Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 227 (2008) 6660–6680
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Derivation and validation of a novel implicit second-order accurate immersed boundary method

Andreas Mark *, Berend G.M. van Wachem

Chalmers University of Technology, Department of Applied Mechanics, 412 96 Gothenburg, Sweden

Received 16 May 2007; received in revised form 17 March 2008; accepted 20 March 2008
Available online 1 April 2008

Abstract

A novel implicit second-order accurate immersed boundary method (IBM) for simulating the flow around arbitrary stationary bodies is developed, implemented and validated in this paper.

The IBM is used to efficiently take into account the existence of bodies within the fluid domain. The flow domain consist of simple Cartesian cells whereas the body can be arbitrary. At the triangulated interface of the body and the fluid, the immersed boundary, the coefficients obtained from discretizing the Navier–Stokes equations are closed with a second-order accurate interpolation arising from the immersed boundary condition employed at the interface. Two different conditions are developed in this paper and it is shown that for the mirroring method the resulting coefficients lead to a well-posed and diagonally dominant system which can be efficiently solved with a preconditioned Krylov sub-space solver. The immersed boundary condition generates a fictitious reversed velocity field inside the immersed boundary, which is excluded from the continuity equation to account for the presence of the IB in the pressure correction equation, resulting in no mass flux over the IB. The force acting on the object from the fluid is determined by integrating the pressure and the viscous forces over the object.

The method is validated by simulating the flow around a sphere for a range of *Re* numbers. It is shown that the drag is very well in agreement with experimental data. Accuracy and convergence studies are employed, proving the second-order accuracy of the method and showing the superiority in convergence rate compared to other IBM. Finally the drag force of a cluster of non-spherical particles is employed to show the generality and potential of the method.

Keywords: Immersed boundary method; Finite-volume method; Implicit method; Immersed boundary condition; Fluid–structure interaction

1. Introduction

Although dispersed multiphase flows are common in both nature and industry, their fundamental behavior is still poorly understood. In recent years, multiphase computational fluid dynamics (CFD) has become an

* Corresponding author. Tel.: +46 31 772 50 34; fax: +46 31 18 09 76.
E-mail addresses: andreas.mark@chalmers.se (A. Mark), berend@chalmers.se (B.G.M. van Wachem).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.03.031

important tool to model and gain more insight into such flows. However, to successfully predict and simulate multiphase flows at a large scale, the fundamental behavior at small scale needs to be understood. The majority of multiphase flows are fluid–solid flows, in which a dispersed solid phase is present in a continuous fluid phase.

Almost all multiphase CFD today is directly aimed at resolving large scales. One of the key issues in successfully describing fluid–solid flows is the physics of the interaction of the two phases. This interaction is a complex process and, to successfully capture it, all the important length and time scales must be resolved; hence true direct numerical simulations (DNS) are required. In true DNS, the full Navier–Stokes equations are solved, taking into account the particle surface, thereby resolving the boundary layer and wake of each particle. In this article, we present a second-order accurate and relatively inexpensive true DNS method to accurately resolve the flow around individual particles.

One of the first methods developed to accurately simulate fluid–solid flows is the arbitrary Lagrangian–Eulerian (ALE) method, proposed by Hu [11,12]. In this method, an unstructured grid is created around the particles and adapted as the particles move. This requires remeshing each time step. Although the method shows good results, the remeshing is computationally very expensive and makes it difficult to provide accurate results. To decrease the computational cost involved by remeshing and to potentially increase the accuracy, three methods without the necessity of remeshing have been employed in the literature: Cartesian, Lagrange Multiplier and Immersed Boundary methods.

In the Cartesian method, or the cell-splitting method [26,33,13,17], the domain mainly consists of Cartesian grid cells. Where a Cartesian cell is cut by the surface segment of a particle, the fluid cells are split and locally unstructured cells are formed. The flow is resolved for the new grid with irregular boundary conditions. The drawback is that the irregular boundary conditions are hard to generalize and small cells can lead to problems with the conservation and stability in the solver.

The Lagrange multiplier method [7,8,6] is a finite-element method where the coupling between the boundary of the particle and the fluid is introduced by a Lagrange multiplier in the integral formulation of the Navier–Stokes equations. The weak coupling is implicitly implemented and, by minimizing the combined integral formulation, the solution of the flow field is found. The method is second-order accurate in space. The drawback with the method is that it is relatively expensive and only preserves the mass and momentum globally, not locally.

The immersed boundary (IB) method resolves these difficulties by using a Cartesian grid in the whole domain. Where the particle surface segments (or immersed boundaries) cross the Cartesian grid cells, the flow variables are directly modified, without the necessity of rearranging the grid. This modification of the flow variables, representing the coupling between the two phases, can be implemented in several different ways. One common way is to exert a Lagrangian force at the immersed boundary onto the fluid phase. The Navier–Stokes equations are then solved on the Cartesian grid, including a number of point forces, representing the influence of the immersed boundaries on the fluid flow.

There exist in general two methods for calculating the forces to represent the immersed boundaries. In the first approach, the forces are calculated from the momentum equations and distributed on the Cartesian mesh by employing a distribution function, first proposed by Peskin [29]; this is called the distributive IB method. The second approach is a non-distributive method, where the Eulerian force is calculated from the momentum equations directly or by employing a boundary condition exactly at the immersed boundary.

The distributive IB method [29,18] employs a function, based on the discrete Dirac delta function, to distribute the Lagrangian forces on the Cartesian mesh. As a result, the Lagrangian forces are smeared out over several grid cells, leading to a smeared representation of the immersed boundary and to only first-order accuracy. In this method, the Lagrangian force is determined explicitly from the generalized Hooke's law.

Goldstein et al. [9] also employs the distributive IB method, but calculates the Lagrangian force by constraining the fluid to the no-slip boundary condition on the immersed boundary. The force proposed by Goldstein et al. [9] contains two unphysical parameters which are dependent on the time scales of the flow. These constants tend to become relatively large, which gives stiff equations to solve. Due to this the method only works well for small time steps. Silva and co-workers [21,22,27] improved the calculation of the Lagrangian forces by making the forces originate from a physical point of view, generally increasing the robustness. The method is explicit and first-order accurate.

To obtain second-order accuracy, Mohd-Yusof [24,25] proposed a non-distributive method, the momentum forcing method. In this method, the no-slip boundary condition is enforced directly at the IB. This is done by introducing a smooth velocity gradient over the sharp IB. To determine the smooth gradient, the velocity field is mirrored over the IB. An explicit momentum force is applied to the interior boundary velocities, such that a linear or bilinear interpolation of the velocity exactly fulfills the no-slip boundary condition at the IB. Due to the reversed velocity field, problems with the mass conservation arise in the boundary cell. The method is employed by, among others, Fadlun et al. [3] and Kang et al. [14].

To deal with the mass conservation errors obtained with the Mohd-Yusof non-distributive method, Kim and Choi [16,15] introduces a mass source/sink in the cells surrounding the IB. This new explicit method shows promising and robust results.

Gilmanov and co-workers [5,4] have also proposed a second-order hybrid Cartesian and IB method, where the IB is triangulated. The external boundary velocities are set by a Dirichlet boundary condition such that an interpolation of the velocity along the normal of the IB fulfills the no-slip boundary condition. The Neumann pressure boundary condition is set in a similar way. The boundary condition for the pressure is applied explicitly, and the boundary condition for the velocity is discretized with first-order accuracy, directly on the exterior grid points. The method shows promising results for low Reynolds numbers.

Majumdar et al. [23] have extended Mohd-Yusof's momentum forcing method to a boundary condition at the IB. To constrain the velocity of the fluid at the IB, a boundary condition is employed exactly at the IB. This boundary condition is formulated either explicitly or implicitly, is second-order accurate in space, and employs three different interpolation schemes. The method has been implemented for stationary IBs along with RANS models to describe turbulent flows. The method shows promising results, but has potential problems with the weighting coefficients in the boundary condition which may result in wiggles in the resulting solution. Moreover, the method generates an unphysical mass flux over the IB.

In [32], Tseng and Ferzinger extended Mohd Yusof's and Fadluns ideas to a ghost-cell immersed boundary method, which extrapolates the fluid flow on a point inside the IB such that the velocity is constrained at the IB, with second-order accuracy. The method shows good results for complex geometries, but an unphysical mass flux over the IB is generated by the method.

The purpose of this paper is to construct, validate and compare two novel stable second-order implicit IB methods. The arbitrary IB is triangulated and in the first method a subset of the triangle vertices are used as control points in which the constraints are applied. A trilinear interpolation is applied to ensure the correct velocity at the IB. Mohd-Yusof's momentum force [25] is rewritten to a fully implicit immersed boundary condition and the generated internal fictitious velocity field is excluded from the continuity equation to account for the presence of the IB in the pressure correction equation, resulting in no mass flux over the IB.

The second method mirrors the velocity along the normal of the triangulated IB segment such that it becomes exactly defined at the IB. Trilinear interpolation is employed to interpolate the velocity to the mirrored velocity point outside the IB.

The present implementation of the methods is done with a staggered variable configuration using a segregated finite volume solver based upon the pressure correction method SIMPLEC [2]. The methods are validated and compared by determining the drag force over a sphere for several different Reynolds numbers. Additionally, the drag force of a cluster of non-spherical particles is simulated to show the generality and potential of the methods.

In the paper, the novel IBM is proven to be second-order accurate in space and to have a superior convergence rate compared to earlier IBMs. The convergence rate is increased because there exists no mass flux over the IB.

2. Governing equations

The equations governing the flow around the immersed boundaries are the continuity and momentum equations for incompressible flow, the Navier–Stokes equations,

$$\frac{\partial u_j}{\partial x_j} = 0, \tag{1}$$

$$\rho\frac{\partial}{\partial t}(u_i)+\rho u_j\frac{\partial u_i}{\partial x_j}=-\frac{\partial p}{\partial x_i}+\frac{\partial}{\partial x_j}\left(\mu\frac{\partial u_i}{\partial x_j}\right)+f_i, \tag{2}$$

where ρ is the density of the flow, u_i is the velocity field of the flow, p is the pressure of the flow, μ represents the viscosity of the fluid, and f_i represents any external source terms. This set of equations is solved together with the immersed boundary condition,

$$u_i = u_i^{ib} \quad \text{on the IB}, \tag{3}$$

which constrains the velocity of the fluid to the velocity of the local IB, u_i^{ib}, exactly at the IB. The immersed boundary condition (IBC) can be implemented in different ways that lead to different methods, which can be both of first- or second-order accurate. A second-order accurate method employs a piece-wise linear representation of Eq. (3); that is, linear interpolation is employed to relate the IB to the surrounding points. In a first-order accurate method, the IB is approximated to the nearest discrete velocity point.

2.1. Discretization of the Navier–Stokes equations

The continuity and momentum equations are discretized on a staggered grid [10,28], which is more natural than a collocated variable arrangement for Cartesian problems, as no artificial boundary conditions are required and spurious modes are prevented. On a staggered grid, the cell faces contain the normal velocity components. The remaining variables, such as density and pressure, are stored at cell centers. A pressure projection method is employed to obtain the correct pressure after an initial guess of the velocity field. For the pressure correction, the SIMPLEC [2] is employed. The SIMPLEC method is a segregated method which first approximates the momentum equation with an estimated pressure field, and then corrects the pressure by employing the continuity equation. Applying this pressure to the previously obtained velocity field results in a velocity field which satisfies the continuity equation, and is close to the momentum equation since the pressure correction term is small. The method iterates until the pressure correction term is very small and hence both the momentum and continuity equations are satisfied.

The SIMPLEC [2] method takes into account the influence from the neighboring velocity corrections in the pressure correction step, instead of neglecting these as in the original SIMPLE [28] method. The magnitude of the velocity corrections is approximated with a weighted mean of velocity neighbors in the pressure and velocity correction equations. These new corrections give a faster convergence in each time step.

3. The immersed boundary condition

The methods developed and employed in this paper are three-dimensional and second-order accurate immersed boundary methods. The methods constrain the velocity of the fluid to a specified velocity exactly at the surface of any body immersed in the fluid flow. The velocity is constrained by an implicitly formulated immersed boundary condition (IBC). In this work, two different discretization schemes of the IBC have been developed. As a result of the application of the IBC, a fictitious velocity field is generated inside the IB. To ensure that there exists no flux over the surface of the IB, a special treatment is adopted to exclude this fictitious velocity field from the governing equations. The surfaces of one or more bodies are triangulated by employing the `gts` library framework [1]. The two different IBC or methods are outlined in the sections below.

3.1. The vertex-constraining IB (VCIB) method

3.1.1. Overview of the method

In the implicit vertex-constraining method, a subset of the vertices of the triangulation are used as control points, $\vec{r}_c$, defining the discretized IB. At these control points, the fluid is constrained to a specified velocity. The subset of vertices is chosen as the set of vertices lying closest to the internal pressure points (cell centers), $\vec{r}_p$; see Fig. 1 for definitions.

To enforce the boundary condition exactly on the immersed boundary, a forcing method, as proposed by Mohd-Yosuf [24], can be employed. In this study, the forcing method is reformulated into an implicit and

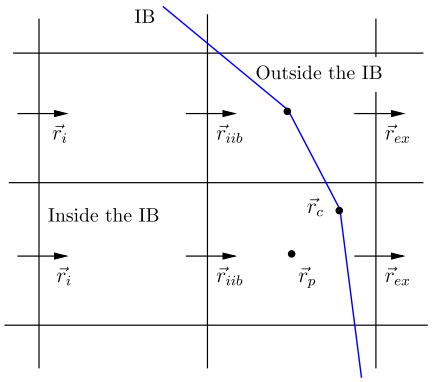

Fig. 1. Definition of a control point, $\vec{r}_c$, which is the vertex in the triangulation of the IB which lies closest to the internal pressure point, $\vec{r}_p$. $\vec{r}_{ex}$ is defined as an external velocity point lying outside the IB in the flow domain, and $\vec{r}_{iib}$ represents the internal immersed boundary (IIB) velocity points lying inside and close to the IB. Finally, a point lying inside the IB but not close to it is defined as an internal velocity point, $\vec{r}_i$. In the figure, the staggered velocity points in the x direction are shown.

second-order accurate framework to an immersed boundary condition. The IBC constrains the velocity of the fluid to the local velocity of the IB, $\vec{u}_{ib}$, exactly at the control point, $\vec{r}_c$. The IBC is employed at the internal immersed boundary (IIB) velocity points inside and close to the IB, $\vec{r}_{iib}$ (see Fig. 1 for definition), such that a trilinear interpolation of the velocity field onto the control point, $\vec{r}_c$, gives the local velocity of the IB, $\vec{u}_{ib}$, at every instant of time due to the implicit formulation of the IBC. As a result of the IBC, the velocity field is reversed over the IB and a fictitious velocity field is generated inside the IB. Due to the fictitious velocity field, a flux over the IB is generated, which is unphysical. Therefore, the fictitious velocity field is excluded in the continuity equation, resulting in no mass flux over the IB. Hence, the presence of the IB is accounted for both in the pressure correction equation and in the momentum equations. The fluid properties are extrapolated onto the triangle centers, which are then used to calculate the surface forces upon the one or more immersed bodies. In [32], a similar boundary condition has been developed but an other extrapolation scheme is employed and the fictitious velocity field inside the IB is included in the continuity equation, resulting in slower convergence rate and flux over the IB.

The VCIB algorithm can be summarized as follows. During initialization, the triangulation, position and velocity of the IBs are read, the velocity points near the triangles obtained from the triangulation of the IB surface are identified, and the corresponding control points, $\vec{r}_c$, are determined. During the assembly of coefficients of the linearized momentum equations, the IBCs at the IB are employed for the velocity points directly adjacent to the triangles, $\vec{r}_{iib}$. The internal velocities defined at the internal velocity points, $\vec{r}_i$, are set to the velocity of the IB. The velocities at the exterior velocity points, $\vec{r}_{ex}$, are set by the linearization and discretization of the Navier–Stokes equations. During the assembly of the pressure correction equation, the velocities at $\vec{r}_{iib}$ are excluded to account for the presence of the IB and to ensure no mass flux over the IB.

3.1.2. Implementation of the immersed boundary condition (IBC)

The immersed boundary condition (IBC) is derived from the Mohd-Yosuf [24] explicit momentum-forcing method. Mohd-Yosuf derive the method by rearranging the discretized momentum equations,

$$a_p u^i_p = \sum_{nb} a_{nb} u^i_{nb} + \Delta p \Delta y \Delta z + f^i \Delta x \Delta y \Delta z,$$

where $u^i(u,v,w)$ are the velocities at the nodes, nb represent the neighboring velocities, and p is the pressure. The discrete momentum forces f^i are only non-zero for boundary velocities and are obtained by replacing the current velocities, u^i_p with the immersed boundary velocities u^i_{ib}, and for the neighboring velocities the old ones are used. This leads to Mohd-Yosuf's explicit momentum forces,

$$f^i \Delta x \Delta y \Delta z = a_p u^i_{ib} - \sum_{nb} a_{nb} \hat{u}^i_{nb} + \Delta p \Delta y \Delta z,$$

where $\hat{u}^i$ are the velocities from the previous time step.

An implicit counterpart is derived by employing the current neighboring velocities directly, instead of calculating the force based upon the velocities and pressure from the previous time step. This implicit method leads to a discretized coefficient condition which can be directly employed as a constraint on the coefficients obtained from linearizing the Navier–Stokes equations. This condition constrains the velocity of the fluid to the velocity of the IB exactly at the IB.

In the rare case that a IIB velocity point coincides with an IB control point, the constraining condition results in a Dirichlet-type boundary condition for the coefficients of the corresponding linearized momentum equation. When the location of the velocity variable does not coincide with an IB control point, the coefficients for the linearized momentum equation are manipulated in such a way that a trilinear interpolation of the velocities onto the control point $\vec{r}_c$ is $\vec{u}_{ib}$.

The following algorithm is employed to apply the immersed boundary condition:

(1) Determine whether velocity and pressure points are inside and near the IB (IIB points), far inside the IB (internal points), or outside the IB (exterior points); see Fig. 1 for definitions. The type of the velocity point is determined and labeled as in Fig. 2.
(2) Find the control point, $\vec{r}_c$ (see Fig. 1), for each pressure cell near the IB.
(3) Calculate the coefficients for the momentum matrices:
   (a) If the coefficients correspond to a IIB velocity point, marked as a 1 in Fig. 2, set the IBC for the points thus:
      (i) Determine which cell is employed for interpolation.
      (ii) Calculate and set the coefficients for the immersed boundary condition (IBC) by trilinear interpolation.
   (b) If the coefficients correspond to an interior velocity node (marked as a 2 in Fig. 2), set the velocity to the local velocity of the IB, $\vec{u}_{ib}$, by employing a Dirichlet boundary condition.
   (c) For all other coefficients, calculate the coefficients from the linearized momentum equations.
(4) Insert the coefficients into the solving matrix and calculate the resulting flow.
(5) Solve the pressure correction equation, based upon the solution in the previous step, excluding the velocities at the IIB velocity points, and correct the pressure and the velocities.
(6) Go to (3) until the continuity equation is fulfilled.
(7) Calculate the surface forces and take the next time step (go to (3)).

The two first steps are done once for a stationary method (i.e the IB does not move) as the points always are of the same type. Further more, it is only necessary to perform steps (a) and (b) for the initializing time step for a stationary IB method, because the coefficients are independent of the solution of the flow problem and

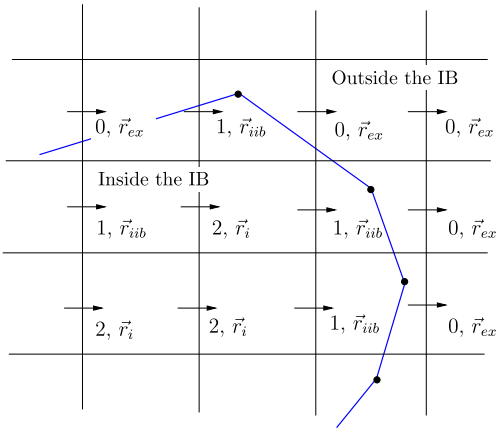

Fig. 2. The markings of the staggered u velocities. The exterior velocity points, $\vec{r}_{ex}$, are marked with a 0; the IIB velocity points, $\vec{r}_{iib}$, are marked with a 1; and the internal velocity points, $\vec{r}_i$, are marked with a 2.

depend only on the geometry of the IB. Coefficients corresponding to the IIB velocity points close to the IB and interior velocities remain unchanged in the assembly of the momentum equations. When the IB is moving, it should be noted that there is a time step restriction due to the linearization of the transient term. Some of the steps are discussed below:

Step 1. *Determine node location relative to the IB.*
The three-dimensional immersed boundary is triangulated by employing the `gts` library framework [1]. By employing the framework, it is determined whether the pressure points and the staggered velocity points lie inside or outside the IB. Velocity points lying within the flow domain, and thus outside the IB, are labeled as in Fig. 2 by a 0. Velocity points lying outside the flow domain and inside the IB domain are labeled with a 1 and a 2 in Fig. 2. The velocity points where at least one of the neighbors lies inside the flow domain (IIB velocity points) are labeled with a 1 in Fig. 2.

Step 2. *Find control points for pressure cells close to the IB.*
For each pressure cell which has at least one neighboring cell whose center lies outside the IB, find the corresponding control point, $\vec{r}_c$. The control point is defined as the closest vertex in the triangulation of the IB, to the current pressure point, $\vec{r}_p$.

Step 3a. *Calculation of the immersed boundary condition.*
The pressure cell whose center, $\vec{r}_p$, lies closest to the IIB velocity point, $\vec{r}_{iib}$, is employed for the interpolation and its control point, $\vec{r}_c$, is employed in the IBC. By determining which pressure cell is employed for interpolation, it is also decided on which side the corresponding IIB velocity point lies, by comparing the Cartesian locations of both points and the control point.Trilinear interpolation is employed to calculate the interpolation coefficients in the IBC. The interpolation is performed in such a way that the fluid velocity is exactly the local velocity of the IB at the control point, $\vec{r}_c(x_c, y_c, z_c)$. The resulting implicitly formulated IBC for the staggered velocity, u, in the x direction in three dimensions follows as:

$$\alpha = \frac{x_{i+1} - x_c}{x_{i+1} - x_i}, \quad \beta = \frac{y_{j+1} - y_c}{y_{j+1} - y_j}, \quad \gamma = \frac{z_{k+1} - z_c}{z_{k+1} - z_k},$$

$$\begin{aligned}
&\alpha\beta\gamma u_{i,j,k} + (1-\alpha)\beta\gamma u_{i+1,j,k} + \alpha(1-\beta)\gamma u_{i,j+1,k} + \alpha\beta(1-\gamma)u_{i,j,k+1} \\
&\quad + (1-\alpha)(1-\beta)\gamma u_{i+1,j+1,k} + (1-\alpha)\beta(1-\gamma)u_{i+1,j,k+1} + \alpha(1-\beta)(1-\gamma)u_{i,j+1,k+1} \\
&\quad + (1-\alpha)(1-\beta)(1-\gamma)u_{i+1,j+1,k+1} = u_{ib},
\end{aligned}$$

where α, β and γ are grid-dependent coefficients. In Fig. A.1 the interpolation cell for the staggered u velocity is shown. This results in a discretization stencil of 8 non-zero coefficients for each IB control point. If the control point, $\vec{r}_c$, lies in the same plane as the velocity node, $u_{i,j,k}$, bilinear or linear interpolation is used. The coefficients for the IIB velocity point are set when the momentum matrix is assembled.

Step 5. *Solving the pressure correction equation.*
It is important that mass is always conserved during flow calculation, and the flow through the IB is precisely zero. The pressure cells directly adjacent to the IB need special treatment such that the mass flux over the IB is exactly zero.
To ensure that no mass flux over the IB exists, the internal face velocities (velocities at the IIB points), having no physical meaning, are excluded when discretizing the continuity equation. By excluding the fictitious velocity field in the pressure correction equation, not only the momentum equation accounts for the presence of the IB, but also the pressure correction equation. Therefore the pressure correction equation generates no driving pressure force over the IB in the momentum equation. Due to this, no regular (Neumann) boundary condition has to be employed at the IB, which previous methods employ to prevent a pressure force over the IB. By employing a pressure BC at the IB unnecessary information is inserted into the pressure correction equations, which could lead to unphysical solutions. Instead the mass conservation over the IB is solved from a physical point of view which generates a more physical solution with a faster convergence rate of the solution, as shown in Section 5.2.

3.2. The mirroring IB (MIB) method

Besides the implementation outlined in the previous section, a different implementation is also employed by implicitly mirroring the velocity over the IB. The implementation mirrors the interior immersed boundary (IIB) node over the IB to a fictitious point in the flow domain. The velocity field at the IIB node is set so that a linear interpolation between that node and the velocity at the mirrored node in the fluid domain, is exactly the velocity of the local segment of the IB. The resulting fictitious velocity field inside the IB is excluded in the continuity equation, so that the mass flux over the IB is zero.

3.2.1. Overview of the method

The MIB method employed in this work directly mirrors the velocity over the IB instead of setting it to the local velocity of the IB, $\vec{u}_{\mathrm{ib}}$, at control points by trilinear interpolation. The internal immersed boundary velocities, $\vec{u}_{\mathrm{iib}}$, are set such that a linear interpolation of the velocity at the IIB point and the interpolated exterior normal velocity, $\vec{u}_{\mathrm{e}}$, is exactly the velocity of the local segment of the IB, $\vec{u}_{\mathrm{ib}}$.

$$\frac{\vec{u}_{\mathrm{iib}} + \vec{u}_{\mathrm{e}}}{2} = \vec{u}_{\mathrm{ib}}.$$

The exterior normal velocity point, $\vec{r}_{\mathrm{e}}$, is the discrete internal immersed boundary velocity point, $\vec{r}_{\mathrm{iib}}$, mirrored along the normal of the plane spanned by its closest triangle (see Fig. 3). Hence the smallest distances between the triangle plane and the fictitious points are the same. From the interior immersed boundary velocity point, $\vec{r}_{\mathrm{iib}}$, a central point on the IB is created, $\vec{r}_{\mathrm{ib}}$, such that the distance between the points is the smallest distance to the IB and therefore normal to the IB.

If the exterior velocity point coincides with a discrete velocity point, the Dirichlet condition for that IIB velocity is trivial. More generally, the exterior normal point lies between the discrete velocity points and therefore the velocity needs to be implicitly interpolated by trilinear interpolation. The IIB velocity can then be set to the reversed interpolated velocity plus the boundary velocity. The interior velocities are set to the boundary (IB) velocity. To ensure that there exists no mass flux over the IB, the same approach as in the first method is adopted.

In previous methods [23] employing techniques similar to mirroring the flow on the IB, wiggles occur in the final solution. We propose that this is due to the presence of a unphysical mass flux and/or a unphysical boundary condition over the IB. To decrease the mass flux over the IB, Majumdar et al. [23] employ a Neumann boundary condition for pressure at the IB. This results in a zero pressure force over the IB, and thus a decreased mass flux. However, a mass flux still exists due to the flow and other forces. The drawback of employing a Neuman boundary condition for the pressure at the IB is a possible ill-conditioned coefficient structure at the IB; the diagonal coefficient can even be zero.

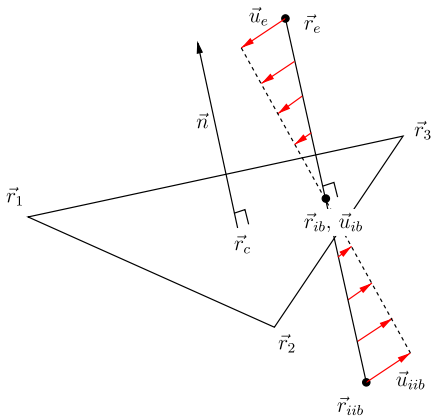

Fig. 3. A triangle in the triangulation with the interior immersed boundary point, $\vec{r}_{\mathrm{iib}}$, the mirrored exterior normal point, $\vec{r}_{\mathrm{e}}$, and the immersed boundary point, $\vec{r}_{\mathrm{ib}}$, and their corresponding velocities $\vec{u}_{\mathrm{e}}$, $\vec{u}_{\mathrm{iib}}$ and $\vec{u}_{\mathrm{ib}}$. $\vec{n}$ is the normal and $\vec{r}_{\mathrm{c}}$ is the centrum of the triangle.

The proposed algorithm can be summarized in the following way: during initialization, the geometry and position of the IBs are read and the types of all velocity points are determined. For each IIB velocity point the corresponding exterior normal velocity point, $\vec{r}_e$, is determined. In the assembly of the coefficients of the linearized momentum equations, the IBCs are employed for the IIB velocity points, the internal velocity points are set to the boundary velocity by Dirichlet boundary conditions, and the exterior velocity points are set by the discretization of the Navier–Stokes equations. The values of the coefficients of the IBCs are determined by first determining the interpolation cell for the exterior normal point and then calculating the coefficients of the implicit trilinear interpolation. Therefore, if the interior immersed boundary velocity, $\vec{u}_{iib}$, at point, $\vec{r}_{iib}$, is appropriately mirrored over the IB, the velocity at the IB is exactly the boundary velocity, $\vec{u}_{ib}$ and the method is inherently stable as the resulting linearized matrix is always diagonally dominant.

3.2.2. Implementation of the immersed boundary condition

The IB condition is a Dirichlet boundary condition which sets the IIB velocity to the reversed exterior normal velocity plus double the boundary velocity. The algorithm employed in the method follows:

(1) Determine whether velocity points lie inside and near the IB (IIB points), far inside the IB (interior points), or outside the IB (exterior points). For definitions of the velocity points, see Fig. 1.
(2) For each velocity inside and close to the IB (IIB velocity), the closest triangle is determined in the triangulation and the exterior normal point, $\vec{r}_e$, mirrored along the normal of the triangle is also determined.
(3) Calculate the coefficients for the momentum matrices.
    (a) If the coefficients correspond to a IIB velocity point, marked as a 1 in Fig. 2, set the IBC for the points as follows:
        (i) Find the trilinear interpolation cell for the exterior normal point, $\vec{r}_e$.
        (ii) Calculate and set the coefficients of the immersed boundary condition (IBC) by interpolation.
    (b) If the coefficients correspond to an interior velocity (marked as a 2 in Fig. 2), set the velocity to the boundary velocity, $\vec{u}_{ib}$, by a Dirichlet boundary condition.
    (c) For all other coefficients calculate the coefficients from linearized momentum equations.
(4) Insert the coefficients into the solving matrix and calculate the resulting flow.
(5) Solve the pressure correction equation, based upon the solution in the previous step, excluding the IIB velocities, and correct the pressure and the velocities.
(6) Go to (3) until the continuity equation is fulfilled.
(7) Calculate the surface forces and take the next time step (go to (3)).

The two first steps are done once for a stationary IB method. As in the first method, it is also only necessary to perform steps (4) and (5) for the initial time step: thus the coefficients are independent of the pressure and velocities, as will be shown below. Hence, in the other time steps the coefficients which correspond to the IIB and interior velocities remain unchanged in the assembly of the momentum matrices. Some of the steps are explained below:

Step 2. *Calculation of the exterior normal point*.
The types of the discrete velocity points are determined as in the first method. The exterior normal point, $\vec{r}_e$, lying on a line from the IIB velocity point, $\vec{r}_{iib}$, normal to the IB, is determined by calculating the closest triangle to the IIB velocity point and the shortest distance to that triangle. The line connecting the IIB velocity point, and the closest point on the triangle, $\vec{r}_{ib}$, points in the normal direction of the triangle $\vec{n}$. Therefore, the exterior normal point, $\vec{r}_e$, and immersed boundary point, $\vec{r}_{ib}$, are determined by

$$\vec{r}_e = \vec{r}_{iib} + 2d\vec{n},$$
$$\vec{r}_{ib} = \vec{r}_{iib} + d\vec{n},$$

where d is the shortest distance between the point, $\vec{r}_{iib}$, and the triangle. See Fig. 4.
Step 3a. *Calculation of immersed boundary condition*.The cell employed for interpolation gives the primed indexes nb'. The immersed boundary condition reads:

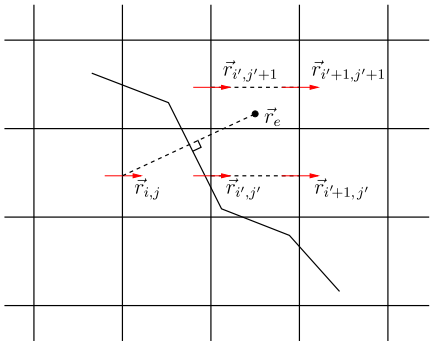

Fig. 4. Visualization in two dimensions of the current discrete IIB velocity point, $\vec{r}_{i,j}$, and the exterior normal velocity point, $\vec{r}_e$, with its surrounding staggered primed indexed velocity points corresponding to the interpolation cell.

$$\frac{\vec{u}_{\mathrm{iib}} + \vec{u}_{\mathrm{e}}}{2} = \vec{u}_{\mathrm{ib}}. \tag{4}$$

Usually the exterior normal point, $\vec{r}_e$, does not coincide with a discrete staggered velocity point and has to be interpolated. As is depicted in Fig. 4, trilinear interpolation is used to implicitly interpolate the velocity to the exterior normal point, $\vec{r}_e$. The equation for this interpolation is then inserted into (4) which gives the discrete IBC for the current IIB velocity $u_{i,j,k}$,

$$\frac{u_{i,j,k} + \sum_{nb'} a_{nb'} u_{nb'}}{2} = \vec{u}_{\mathrm{ib}}, \tag{5}$$

where $a_{nb'}$ are the interpolation coefficients for the neighboring velocity points. The coefficient in front of the current velocity, $u_{i,j,k}$, is always larger than or equal to any of the other coefficients, leading to a diagonally dominant immersed boundary coefficient condition. There exists only one case in which another coefficient is equal, namely when $\vec{r}_e$ coincides with a discrete staggered velocity point, $\vec{r}_{i,j}$. But this leads only to a simple Dirichlet boundary condition equation (4) and this equation is also well posed. In the next section, the diagonal dominance of the linearized system including the IBC will be shown.

3.2.3. Diagonal dominance

From Eq. (5) and the scalar coefficients from the trilinear interpolation, the diagonal coefficient always lies in the following domain:

$$1 \leqslant a_{i,j,k} \leqslant 1 + \alpha\beta\gamma,$$

where α, β, γ are scalars from the interpolation. The trilinear interpolation coefficients $a_{nb'}$, corresponding to the implicit interpolation of the exterior fictitious. point, are always positive and the maximum of them lies in

$$0 \leqslant \max_{nb'}(a_{nb'}) \leqslant 1,$$

where $\max(a_{nb'})$ is only equal to one if $\vec{r}_e$ coincides with a discrete velocity point $\vec{r}_{\mathrm{iib}}$. Therefore,

$$0 \leqslant \max(a_{nb'}) \leqslant 1 \leqslant a_{i,j,k} \leqslant 1 + \alpha\beta\gamma.$$

From this it can be concluded that the coefficient system obtained from the IBC is always well posed or the corresponding rows in the matrix are always diagonally dominant. Therefore the method gives a strong and robust formulation.

4. Calculation of the surface forces

In most IB methods, the fluid velocity and pressure at the IB are controlled by distributive forces. In such cases, the total force on the fluid as a result of the IB is simply the sum of the distributive forces applied to each

IB element. Since in the current method the flow properties are controlled in the discretization itself, such a force is not explicitly calculated, but often this force is required. The total force is given by the surface integral of the stress tensor over the IB,

$$F_i = \int_{\mathrm{IB}} (-p\delta_{ij} + \tau_{ij}) n_j \,\mathrm{d}S = \int_{\mathrm{IB}} \left(-p\delta_{ij} - \mu \left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right] \right) n_j \,\mathrm{d}S. \tag{6}$$

The total surface force has two parts, the pressure force and the viscous force. These forces are integrated over the surface of the IB. The discretization of the integral is determined for each triangle by extrapolation and summed up to get the total surface force acting on the IB. To calculate the pressure force, the pressure is extrapolated onto the center of the triangle. This is done by defining two exterior points $(\vec{r}', \vec{r}'')$ which lie on the normal of the triangle. $\vec{r}'$ has twice the distance to $\vec{r}_{\mathrm{c}}$ as $\vec{r}''$. The pressure is then interpolated onto the exterior points by trilinear interpolation where only exterior pressure points are used. The pressure part of the surface force is discretized and calculated as (see Fig. 5)

$$F_i^{\mathrm{press}} = \int_{\mathrm{IB}} -p\delta_{ij} n_j \,\mathrm{d}S = \sum_k^{k\in\mathrm{IB}} -p_{\mathrm{c}}^k n_i^k \,\mathrm{d}S^k. \tag{7}$$

In order to calculate the viscous force, the tensor τ_{ij} needs to be determined at the center of the triangle. This is done by introducing three new points, which are the projections of an exterior extrapolation point onto the Cartesian axes. The velocities are then interpolated onto these points by trilinear interpolation. The viscous force in the x direction becomes (see Fig. 6)

$$\begin{aligned} F_x^{\mathrm{visc}} &= \int_{\mathrm{IB}} \tau_{xj} n_j \,\mathrm{d}S = \int_{\mathrm{IB}} -\mu \left(\frac{\partial u}{\partial x_j} + \frac{\partial u_j}{\partial x} \right) n_j \,\mathrm{d}S \\ &= -\mu \sum_k^{k\in\mathrm{IB}} \left(\frac{\vec{u}_{\vec{r}_x} - \vec{u}_{\vec{r}_{\mathrm{c}}}}{\vec{n}_x} \vec{n}_x + \frac{\vec{u}_{\vec{r}_y} - \vec{u}_{\vec{r}_{\mathrm{c}}}}{\vec{n}_y} \vec{n}_y + \frac{\vec{u}_{\vec{r}_z} - \vec{u}_{\vec{r}_{\mathrm{c}}}}{\vec{n}_z} \vec{n}_z + \frac{\vec{u}_{\vec{r}_x} - \vec{u}_{\vec{r}_{\mathrm{c}}}}{\vec{n}_x} \vec{n}_x + \frac{\vec{v}_{\vec{r}_x} - \vec{v}_{\vec{r}_{\mathrm{c}}}}{\vec{n}_x} \vec{n}_y + \frac{\vec{w}_{\vec{r}_x} - \vec{w}_{\vec{r}_{\mathrm{c}}}}{\vec{n}_x} \vec{n}_z \right) \mathrm{d}S^k \\ &= -\mu \sum_k^{k\in\mathrm{IB}} \left(\vec{u}_{\vec{r}_x} + \vec{u}_{\vec{r}_y} + \vec{u}_{\vec{r}_z} - 3\vec{u}_{\vec{r}_{\mathrm{c}}} + (\vec{u}_{\vec{r}_x} - \vec{u}_{\vec{r}_{\mathrm{c}}}) + (\vec{v}_{\vec{r}_x} - \vec{v}_{\vec{r}_{\mathrm{c}}}) \frac{\vec{n}_y}{\vec{n}_x} + (\vec{w}_{\vec{r}_x} - \vec{w}_{\vec{r}_{\mathrm{c}}}) \frac{\vec{n}_z}{\vec{n}_x} \right) \mathrm{d}S^k \end{aligned} \tag{8}$$

and the other directions follow similarly. If a normal is zero in a certain direction, the resulting term gives no contribution to the total shear force.

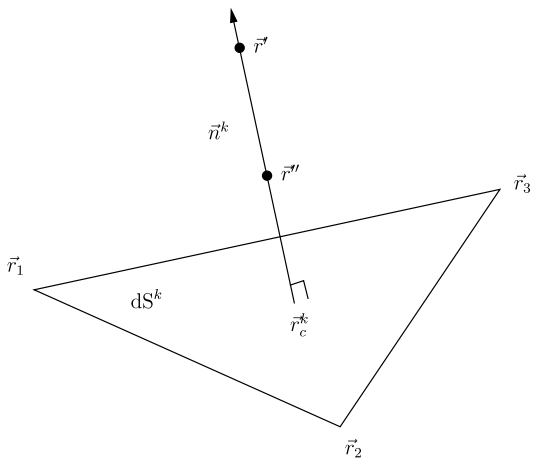

Fig. 5. Triangle k in the triangulation of the immersed boundary with the exterior points, $\vec{r}'$ and $\vec{r}''$. $\mathrm{d}S^k$ is the area and $\vec{n}^k$ is the normal of the triangle.

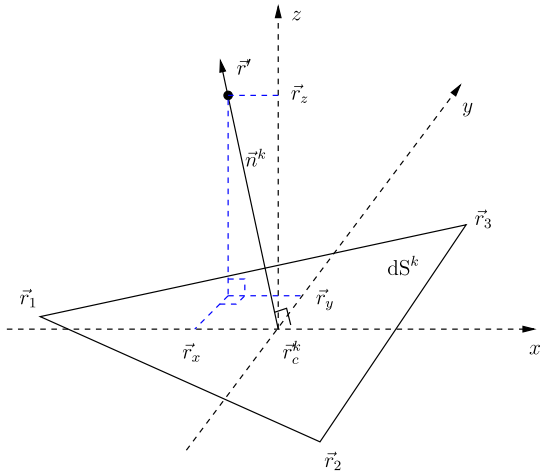

Fig. 6. Triangle k in the triangulation of the immersed boundary with the projection of $\vec{r}_p$ onto the Cartesian coordinate axes $(\vec{r}_x, \vec{r}_y, \vec{r}_z)$.

5. Results and discussion

5.1. Numerical simulation of a sphere immersed in a fluid

To validate the derived IB methods, the flow around a unit sphere at various Reynolds numbers is simulated. For the flow over a sphere at low Reynolds numbers $Re \ll 1.0$, the Stokes drag force is given by

$$f_d = f_{visc} + f_{press} = 4\pi\mu R v_\infty + 2\pi\mu R v_\infty,$$

where R is the radius of the sphere, μ the viscosity of the fluid and v_∞ is the free stream velocity. From this, the Stokes drag coefficient can be derived,

$$C_d = \frac{24}{Re}.$$

At higher Reynolds numbers, semi-empirical equations are used to validate the method, Table 1 [30].

The Cartesian grid has $100 \times 100 \times 100$ computational cells with a constant grid size. The grid size ranges from 0.2 m to 0.05 m, depending on the Reynolds number. For high Reynolds numbers, a small grid size is employed to resolve the small flow scales around the sphere. The time step and grid size in the simulations are chosen such that a constant CFL number of 0.1 is employed. The density of the fluid, ρ_f, is 1.0 kg/m^3 and the viscosity, μ, is 0.1 N s/m^2. The radius of the sphere is 0.5 m and the sphere is fixed in the center of the computational domain. In the inlet and on the side walls, the velocity of the fluid is set to the mean stream velocity U_∞ and in the outlet a Neumann boundary condition is employed. The fluid pressure is set to zero in the outlet and a Neumann pressure boundary condition is employed where velocity is specified. The same fluid boundary conditions are employed throughout this paper.

Table 1
The drag on a sphere immersed in a fluid

References	Range	Relationship for C_d
Stoke	$Re \ll 1$	$\frac{24}{Re}$
Shiller and Nauman [31]	$Re < 800$	$\frac{24}{Re}(1 + 0.15Re^{0.687})$
Lapple [20]	$Re < 1000$	$\frac{24}{Re}(1 + 0.125Re^{0.72})$
Langmuir and Blodgett [19]	$1 < Re < 100$	$\frac{24}{Re}(1 + 0.197Re^{0.63} + 2.6 \times 10^{-4}Re^{1.38})$

Various authors have proposed relations for varying Reynolds number ranges.

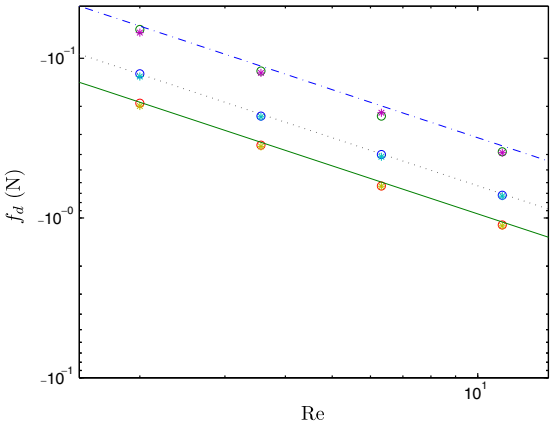

Fig. 7. The total Stokes drag force, f_d (—), viscous force, f_{visc} (·····), and pressure force, f_{press} (-·-·-·-·-), are plotted against the simulated forces (VCIB "○" and MIB "*"). From the figure it can be concluded that the simulations accurately reproduces the drag force for *Re* less than one. The drag force is slightly low due to the influence of the sliding wall.

In Fig. 7 the simulated total drag force, f_d, viscous force, f_{visc}, and pressure force, f_{press}, are plotted against the result obtained by Stokes for Reynolds numbers less than one. From the figure it can be concluded that both the vertex-constraining immersed boundary (VCIB) method, and the mirroring immersed boundary (MIB) method simulate the drag force correctly for *Re* less than one.

To show that the methods are valid for higher Reynolds numbers, the known relationships for sphere drag and the simulated drag coefficient, C_d, for both methods are plotted in Fig. 8. It can be seen that the simulated drag coefficients lie very close to the established relations. For the highest Reynolds numbers the drag coefficient differs slightly from the theory because the domain size starts to affect the flow and the known drag relationships are determined for spheres immersed in an infinite fluid. Because of numerical instabilities encountered when employing the vertex-constraining immersed boundary (VCIB) method, we have not succeeded in employing this method for flows having higher Reynolds numbers. In Table 2 the simulation data for both methods along with the drag predictions of the relations shown in Table 1 are displayed.

In Fig. 9 a simulation of a sphere at Reynolds number 500 is visualized. The instantaneous stream lines and pressure contours of the flow around the sphere are shown to visualize the properties of the unsteady wake behind the sphere.

To further show that the viscous and the pressure parts of the surface force are correctly simulated, a grid refinement study of the skin friction, C_f, and the pressure coefficient, C_p, is performed. The dimensions of the

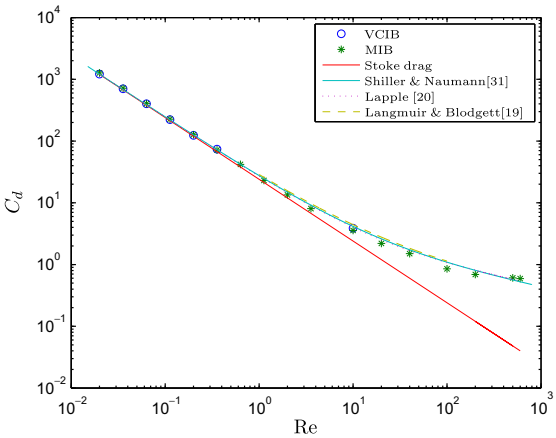

Fig. 8. The simulated drag coefficient, C_d, for both methods displayed with the known relationships for sphere drag, Table 1.

Table 2
Simulated drag coefficients as a function of the Reynolds number for the VCIB and MIB methods compared to a number of relation (Table 1)

Re	VCIB	MIB	Stoke	Shiller and Nauman [31]	Lapple [20]	Langmuir and Blodgett [19]
0.020	1215.9	1266.9	1200.0	1212.2	1209.0	–
0.037	703.2	715.3	674.1	684.4	681.8	–
0.063	401.6	403.3	379.7	388.3	386.2	–
0.112	222.7	225.7	213.5	220.7	219.0	–
0.200	123.5	126.7	120.0	126.0	124.7	–
0.356	73.9	71.7	67.4	72.4	71.4	–
0.632	–	41.8	37.9	42.1	41.4	–
1.12	–	23.0	–	24.8	24.3	25.9
2.00	–	13.4	–	14.9	14.5	15.7
3.56	–	8.04	–	9.16	8.84	9.71
10	3.87	3.57	–	4.15	3.97	4.43
20	–	2.19	–	2.61	2.50	2.78
40	–	1.50	–	1.67	1.73	1.83
100	–	0.853	–	1.09	1.07	–
200	–	0.690	–	0.806	0.801	–
500	–	0.608	–	0.563	0.575	–
600	–	0.591	–	0.526	0.540	–

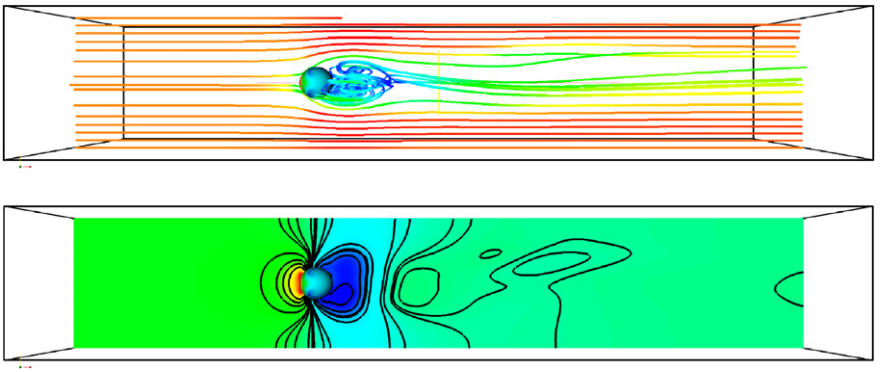

Fig. 9. Simulation of a sphere immersed in a fluid at *Re* numbers 500. Top: The instantaneous stream lines colored by the velocity show the properties of the unsteady wake behind the sphere. Bottom: The pressure is visualized on a plane through the 3D simulation domain along with some pressure contours. Red indicates high velocity or pressure.

computational domain employed in the study are (2.4 m × 2.4 m × 2.4 m) and a sphere with diameter 1.0 m is placed in the center of the domain. The viscosity of the fluid is 0.01 N s/m^2 resulting in a Reynolds number of 150 based on the sphere diameter and the fluid density set to unity. In the study, the number of computational cells in each direction for the whole computational domain is: 100, 80, 60, 48, 40, 30. In Fig. 10 the simulated coefficients for the different grid resolutions at $t = 5.0$ s are shown in a plane with the normal in the z direction. There is a small difference between the simulations at the front of the sphere and at maximum amplitude of the skin friction as the boundary layer is not completely resolved with the coarse computational domain. But no oscillations of the skin friction are observed such as previous methods with even finer grid sizes generate. Therefore it is concluded that the surface forces are overall accurately simulated.

5.2. Convergence rate of the MIB method

To confirm that the convergence rate increases as the fictitious velocity field inside the IB is excluded in the pressure correction equation, a convergence study of the MIB method is performed. In the convergence study a sphere immersed in a fluid is simulated and the velocity at the surface of the sphere is set. The computational

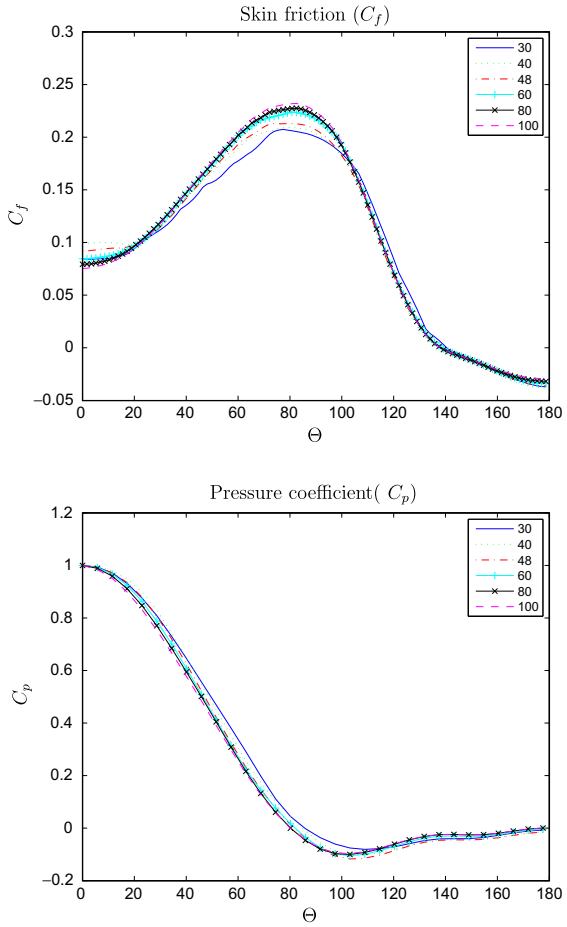

Fig. 10. Top: The skin friction for different grid sizes. Bottom: Pressure coefficient for different grid sizes. (—, ····, -·-·-·-, -+-, -×-, - - - - -) 30, 40, 48, 60, 80, 100.

Table 3
Number of iterations required until convergence is obtained in the first time step if the fictitious velocity field is excluded or included in the pressure correction equation

Re	It exclude	It include	Difference (%)
200	11	31	64.5
100	16	29	44.8
10	14	27	48.2
1	13	19	31.2

domain employed in the study has $28 \times 32 \times 32$ computational cells with a constant width of 0.05 m. The time step employed in the simulation corresponds to a constant CFL number of 0.01. The density of the fluid is set to unity, the viscosity is 0.01 N s/m^2 and the radius of the sphere is 0.2 m.

The convergence study is performed with four different Reynolds numbers and the number of iterations employed in the first time step until convergence, when the fictitious velocity field is excluded (MIB method) and included in the pressure correction equation, is shown in Table 3. Other IBMs [23,32] include the fictitious velocity field in the pressure correction equation, resulting in a mass flow over the IB. From the table it is concluded that excluding the fictitious velocity field in the pressure correction equation increases the convergence rate up to 60% and that the difference between the methods increases with the Reynolds number. In Fig. 11 the

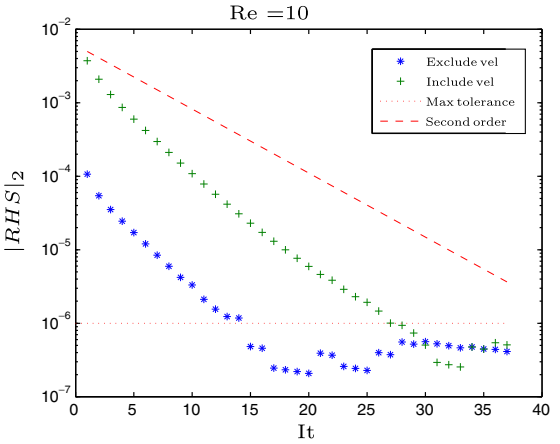

Fig. 11. The 2-norm of the total error in the continuity equation is shown for each iteration step at the Reynolds number 10. '+' represents the simulation when the velocity field inside the IB is included and '*' where it is excluded (MIB method) in the pressure correction equation.

2-norm of the total error in the continuity equation is shown for each iteration step at the Reynolds number 10. The figure shows that the error in the first time step differs by two orders of magnitude. As a result of the convergence study it is concluded that the new proposed method increases the convergence rate by properly excluding the fictitious velocity field in the continuity equation, therefore ensuring no mass flux over the IB and preventing oscillations in the solution.

5.3. Accuracy of the MIB method

To validate the accuracy of the MIB method, the drag coefficient, C_d, of a sphere immersed in a fluid is simulated for several different mesh sizes and Reynolds numbers. The sphere employed in the simulations has a diameter, D, of 0.4 m. The fluid density is 1.0 kg/m^3 and the viscosity is 0.01 N s/m^2. The different computational domains, mesh sizes and time steps employed in the simulations are shown in Tables 4 and 5.

Table 4
The different computational domains and mesh sizes employed in the simulation of the flow around a single sphere

$N \times N \times N$	Δx	$D/\Delta x$
$10 \times 10 \times 10$	0.200	2.0
$16 \times 16 \times 16$	0.125	3.2
$20 \times 20 \times 20$	0.100	4.0
$25 \times 25 \times 25$	0.080	5.0
$40 \times 40 \times 40$	0.050	8.0

Table 5
The different time steps employed for different computational domains in the simulations

$Re = 0.1$		$Re = 1$		$Re = 10$		$Re = 100$	
$N \times N \times N$	t	$N \times N \times N$	t	$N \times N \times N$	t	$N \times N \times N$	t
$10 \times 10 \times 10$	0.01	$10 \times 10 \times 10$	0.0005	$10 \times 10 \times 10$	0.0001	$10 \times 10 \times 10$	0.002
$16 \times 16 \times 16$	0.01	$16 \times 16 \times 16$	0.0010	$16 \times 16 \times 16$	0.0010	$16 \times 16 \times 16$	0.002
$20 \times 20 \times 20$	0.01	$20 \times 20 \times 20$	0.0020	$20 \times 20 \times 20$	0.0010	$20 \times 20 \times 20$	0.002
$25 \times 25 \times 25$	0.01	$25 \times 25 \times 25$	0.0100	$25 \times 25 \times 25$	0.0010	$40 \times 40 \times 40$	0.002
$40 \times 40 \times 40$	0.01	$40 \times 40 \times 40$	0.0100	$40 \times 40 \times 40$	0.0010		

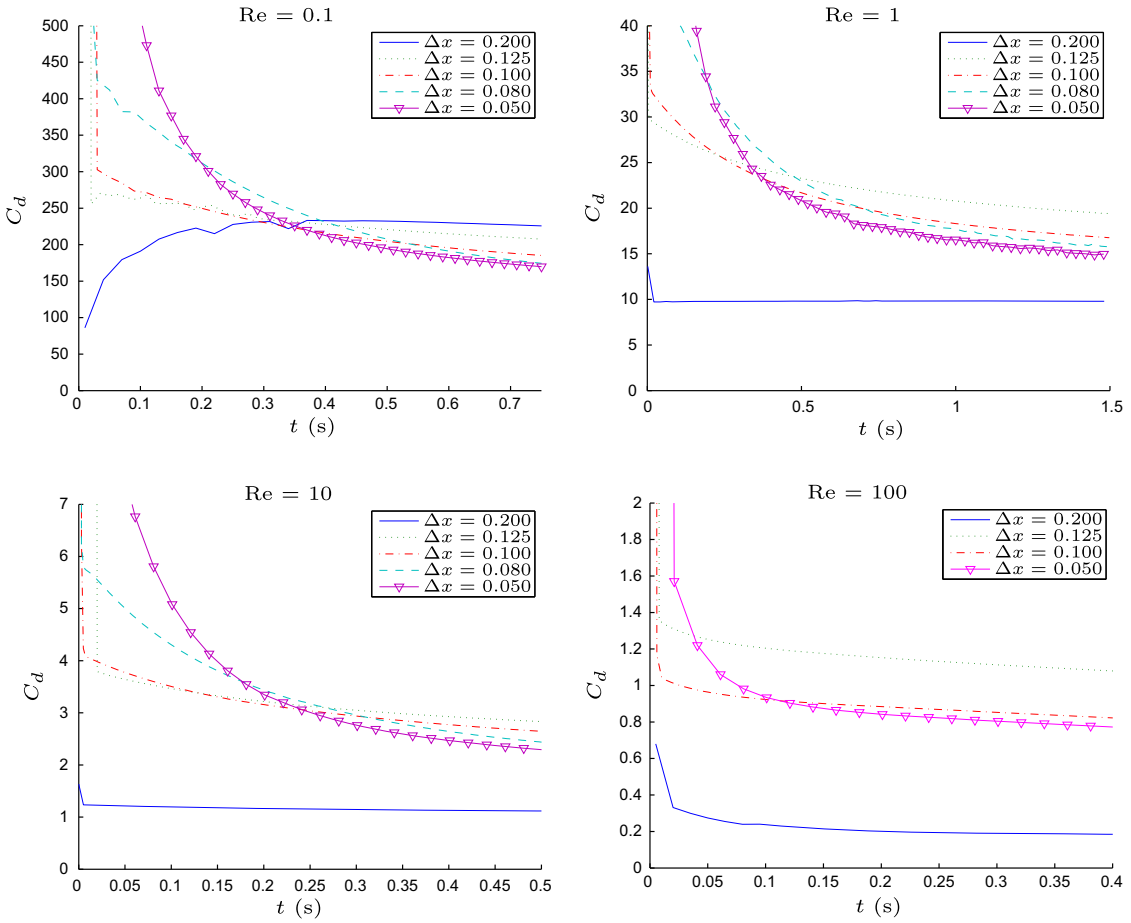

Fig. 12. The simulated drag coefficient as a function of time for the four different Reynolds numbers and five different mesh spacings.

Fig. 12 shows the simulated drag coefficient as a function of time for four Reynolds numbers and various mesh spacings. The solutions show almost no fluctuations and it can be concluded that the MIB method has good temporal accuracy. The relative error for different grid sizes is shown in Fig. 13, where the relative error is calculated as $|C_d - C_d^*|$, in which C_d is the simulated drag coefficient and C_d^* is the simulated drag coefficient on the finest grid. In the figure two lines are drawn with a first-order and a second-order slope and from these lines it can be seen that the method is at least second-order accurate. To calculate the order of accuracy of the MIB method for each Reynolds number, linear regression is employed on the results and the resulting order of accuracy is shown in Table 6. From the table it is concluded that the MIB method is at least second-order accurate.

5.4. Numerical simulation of a cluster of non-spherical particles immersed in a fluid

Finally the drag force of a cluster of non-spherical particles is simulated to show the generality and potential of the method. In Fig. 14, the simulation of a cluster with 10 particles is shown along with the pressure distribution on the surfaces of the particles and stream lines following the fluid flow. From the pressure distribution, it can be seen how the position of each particle affects the drag force acting upon the particles. As a result, a particle lying in the wake of another particle has a lower drag force than the particle in front. Such simulations will be used to develop large-scale drag models for clusters of particles.

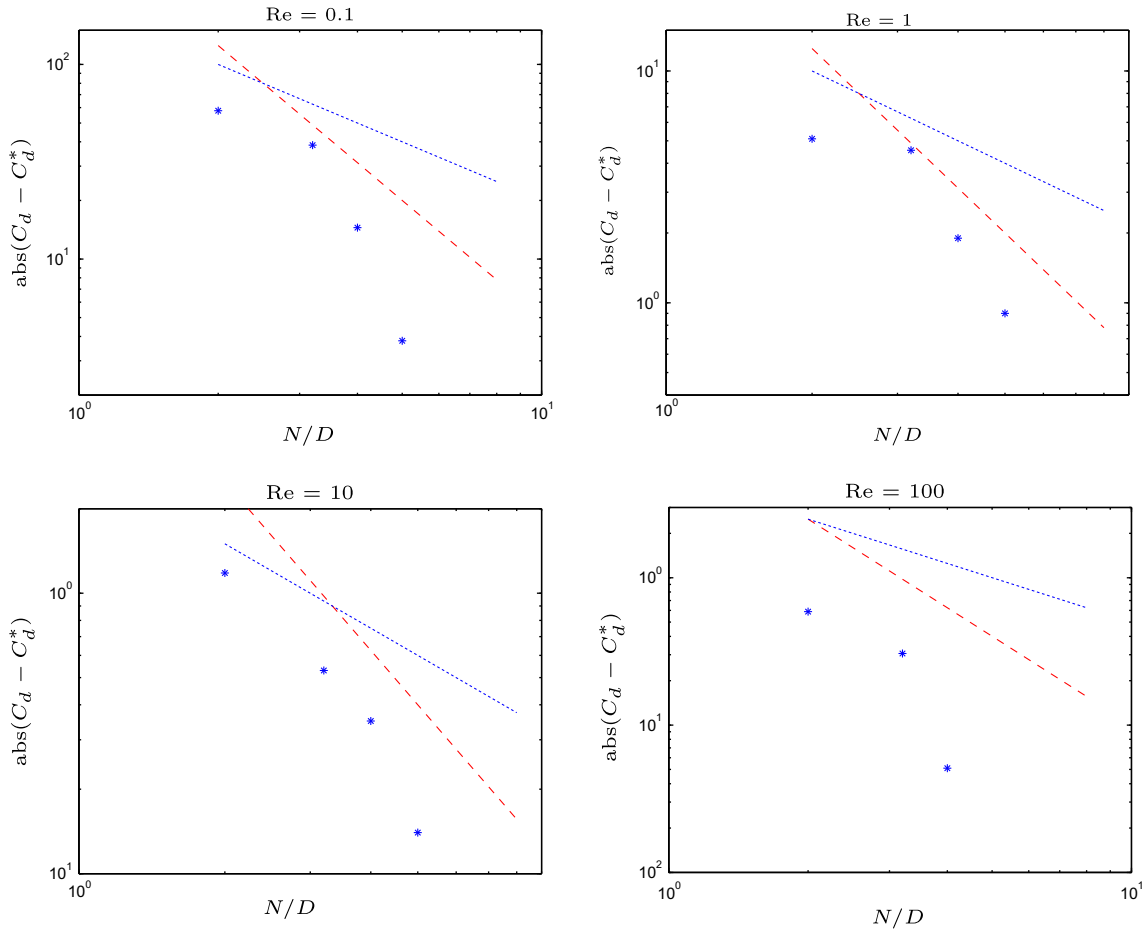

Fig. 13. The relative error in the computed drag coefficient as a function of the mesh spacing, N/D. First-order accuracy (····) and second-order accuracy (-------) are shown for reference.

Table 6
Order of accuracy of the MIB method as a function of Reynolds numbers

Re	Order of accuracy
0.1	−2.8
1	−1.9
10	−2.2
100	−3.2

5.5. Comparison of methods

It has been shown that the mirroring immersed boundary (MIB) method is numerically more stable than the vertex-constraining immersed boundary (VCIB) method, because the matrix resulting from the linearization of the Navier–Stokes equations along with the immersed boundary condition is always diagonally dominant. In the vertex-constraining method, occasionally rows are not diagonally dominant, leading to a difficult system to solve. The mirroring (MIB) method gives faster convergence and provides stable results for any Reynolds number.

The mirroring method is also independent of the number of vertices in the triangulation, because it uses control planes, spanned by the closest triangles, instead of control points. Therefore, the mirroring method

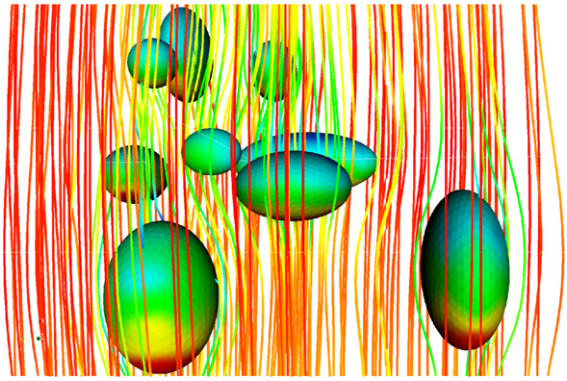

Fig. 14. A simulation of a cluster of non-spherical particles of various sizes. The pressure distribution is visualized on the surface of the particles and the stream lines show the fluid flow.

can track more general IBs and the size of the triangles of the triangulation does not play a role. Further, in the VCIB method, problems with convex surfaces can arise, as a large number of discrete velocity points employed in the extrapolation lies inside the IB.

6. Conclusions

In this paper, a novel implicit second-order accurate immersed boundary method for simulating the flow around stationary arbitrary bodies is developed, implemented and validated. The velocity of the fluid is constrained by an implicit immersed boundary condition such that it exactly follows the immersed boundary. Two such conditions are developed, validated and compared, the vertex-constraining (VCIB) method which constrains the velocity at vertex control points and the mirroring (MIB) method which mirrors the velocity field over the immersed boundary.

From simulations of the flow around a sphere, it is shown that both methods correctly simulate the fluid flow around a sphere and are second-order accurate. From the skin friction and pressure coefficient it is shown that no oscillations in the surface force are generated as in previous methods. The difference between the methods lies in the convergence rate and stability, where the mirroring method is more stable and has faster convergence. By performing a convergence study it is shown that, by excluding the fictitious velocity field in the pressure correction equation, the convergence rate increases significantly. A grid refinement study is performed to show that the MIB method is at least second-order accurate in space. Finally, the mirroring immersed boundary method is employed to resolve the flow around 10 non-spherical particles immersed in a fluid.

Acknowledgment

This work was financed by the Swedish Research Council (VR), Grant No. 621-2004-2008.

Appendix A. Trilinear Interpolation

In the proposed IB methods, interpolation of various variables in the three-dimensional discretized space is frequently required. To interpolate a property to a given point in the discretized space, trilinear interpolation is employed. In trilinear interpolation, the properties of the eight corners of an interpolation box are linearly weighted onto the control point to obtain the interpolated value. The interpolation of the staggered velocity u onto the control point $\vec{r}_c$ is shown as an example; see Fig. A.1. Given the scaling factors

$$\alpha = \frac{x_{i+1} - x_c}{x_{i+1} - x_i}, \quad \beta = \frac{y_{j+1} - y_c}{y_{j+1} - y_j}, \quad \gamma = \frac{z_{k+1} - z_c}{z_{k+1} - z_k}$$

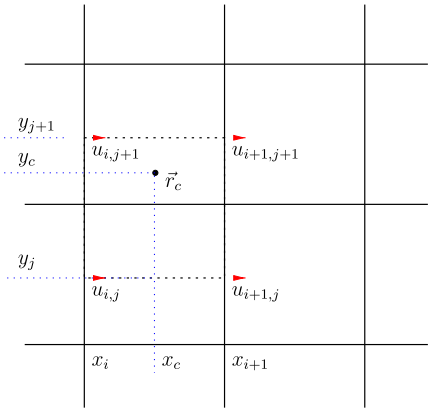

Fig. A.1. A two-dimensional trilinear interpolation cell for the staggered velocities u onto the control point, $\vec{r}_c$; the dimension in the z direction follows analogy.

the velocity at point $\vec{r}_c$ can be expressed as

$$u_{\vec{r}_c} = \alpha\beta\gamma u_{i,j,k} + (1-\alpha)\beta\gamma u_{i+1,j,k} + \alpha(1-\beta)\gamma u_{i,j+1,k} + \alpha\beta(1-\gamma)u_{i,j,k+1} + (1-\alpha)(1-\beta)\gamma u_{i+1,j+1,k} + (1-\alpha)\beta(1-\gamma)u_{i+1,j,k+1} + \alpha(1-\beta)(1-\gamma)u_{i,j+1,k+1} + (1-\alpha)(1-\beta)(1-\gamma)u_{i+1,j+1,k+1}. \quad (A.1)$$

Different sets of interpolation factors α, β, γ are determined for different types of variables (staggered or non-staggered) and different control points.

Appendix B. Triangle properties

Triangle k in the triangulation of the IB is defined by three points $\vec{r}_1$, $\vec{r}_2$ and $\vec{r}_3$. The vectors $p_1\vec{p}_2$ and $p_1\vec{p}_3$ spans the plane of the triangle (see Fig. B.1). Therefore the normal of the plane can be calculated by

$$\vec{n}^k = \frac{p_1\vec{p}_2 \times p_1\vec{p}_2}{|p_1\vec{p}_2 \times p_1\vec{p}_2|} \quad (B.1)$$

and the area (surface) by

$$\mathrm{d}S^k = \frac{1}{2}|p_1\vec{p}_2 \times p_1\vec{p}_2|. \quad (B.2)$$

Further, the center of the triangle is calculated from

$$\vec{r}_c^{\,k} = \frac{1}{3}\sum_{i=1}^{3}\vec{r}_i. \quad (B.3)$$

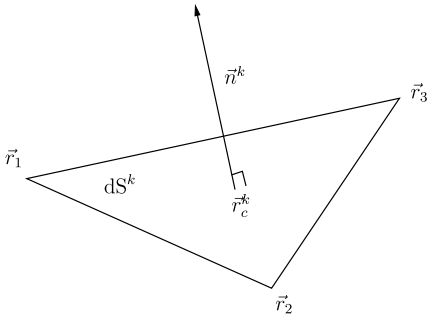

Fig. B.1. A triangle of the immersed boundary, with the points, $\vec{r}_i$, the center, $\vec{r}_c^{\,k}$, normal, $\vec{n}^k$, and area, $\mathrm{d}S^k$, of triangle k.

References

[1] GTS home page, http://gts.sourceforge.net/, 2000.
[2] J.P. Van Doormaal, G.D. Raithby, Enhancements of the simple method for predicting incompressible fluid flows, Numer. Heat Transf. 7 (1984) 147–163.
[3] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35–60.
[4] A. Gilmanov, F. Sotiropoulos, A hybrid Cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies, J. Comput. Phys. 207 (2005) 457–492.
[5] A. Gilmanov, F. Sotiropoulos, E. Balaras, A general reconstruction algorithm for simulating flows with complex 3D immersed boundaries on Cartesian grids, J. Comput. Phys. 191 (2003) 660–669.
[6] R. Glowinski, T.-W. Pan, T.I. Hesla, D.D. Jospeh, A distributed Lagrange multiplier/fictitious domain method for particulate flows, Int. J. Multiphase Flow 25 (1999) 755–794.
[7] R. Glowinski, T.-W. Pan, J. Periaux, A fictitious domain method for Dirichlet problem and applications, Comput. Methods Appl. Mech. Eng. 111 (1994) 283–303.
[8] R. Glowinski, T.-W. Pan, J. Periaux, A fictitious domain method for external incompressible viscous flow modeled by Navier–Stokes equations, Comput. Methods Appl. Mech. Eng. 112 (1994) 133–148.
[9] D. Goldstein, R. Handler, L. Sirovich, Modeling a non-slip flow boundary with an external force field, J. Comput. Phys. 105 (1993) 354–366.
[10] F.H. Harlow, A.A. Amsden, Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface, Phys. Fluids 8 (1965) 2182–2189.
[11] H.H. Hu, Direct simulation of flows of solid–liquid mixtures, Int. J. Multiphase Flow 22 (2) (1996) 335–352.
[12] H.H. Hu, N.A. Patankar, M.Y. Zhu, Direct numerical simulation of fluid–solid systems using arbitrary Lagrangian–Eulerian technique, J. Comput. Phys. 169 (2001) 427–462.
[13] G. Kalatzin, G. Iaccarino, Toward immersed boundary simulation of high reynolds number flows, Technical Report, Center of Turbulence Research, Annual Research Briefs, 2003.
[14] S. Kang, G. Iaccarino, P. Moin, Accurate and efficient immersed-boundary interpolations for viscous flows, Technical Report, Center for Turbulence Research, Annual Research Brief, 2004.
[15] D. Kim, H. Choi, Immersed boundary method for flow around an arbitrarily moving body, J. Comput. Phys. 212 (2006) 662–680.
[16] J. Kim, D. Kim, H. Choi, An immersed-boundary finite-volume method for simulations of flow in complex geometries, J. Comput. Phys. 171 (2001) 132–150.
[17] M.P. Kirkpatrick, S.W. Armfield, J.H. Kent, A representation of curved boundaries for the solution of the Navier–Stokes equations on a staggered three-dimensional Cartesian grid, J. Comput. Phys. 184 (2003) 1–36.
[18] M.-C. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.
[19] I. Langmuir, K.B. Blodgett, U.S. Army Airforce Technical Report, 1948, p. 5418.
[20] C.E. Lapple, Particle dynamics, Ver. Deut. Ing. (1951).
[21] A.L.F. Lima E Silva, A. Silveira-Neto, J.J.R. Damasceno, Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method, J. Comput. Phys. 189 (2003) 351–370.
[22] A.L.F. Lima E Silva, A. Silveira-Neto, J.J.R. Damasceno, Numerical simulation of two-dimensional complex flows over bluff bodies using the immersed boundary method, Technical Report, School of Mechanical Engineering, Federal University of Uberlandia, Brazil, 2004.
[23] S. Majumdar, G. Iaccarino, P. Durbin, Rans solvers with adaptive structured boundary non-conforming grids, Technical Report, Center for Turbulence Research, Annual Research Briefs, 2001.
[24] J. Mohd-Yusof, Combined immersed-boundary/b-spline methods for simulations of flow in complex geometries, Technical Report, Center for Turbulence Research, Annual Research Briefs, 1997.
[25] J. Mohd-Yusof, Development of immersed boundary methods for complex geometries, Technical Report, Center for Turbulence Research, Annual Research Briefs, 1998.
[26] W.F. Noh, Cel: a time-dependent, two-space-dimensional, coupled Eulerian–Lagrange code, Methods Comput. Phys. (1964) 117–179.
[27] J.E.S. Oliveira, A.L.F. Lima E Silva, A. Silveira-Neto, Numerical simulation of high reynolds number flows over circular cylinders using the immersed boundary method with turbulence modelling, Technical Report, Mechanical Engineering College, 2005.
[28] S.V. Patankar, Numerical Heat Transfer and Fluid Flow, Hemisphere Publishing Corporation, 1980.
[29] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[30] J.R. Grace, R. Clift, M.E. Weber, Bubbles, Drops and Particles, Academic Press, London, 1978.
[31] L. Shiller, A.Z. Naumann, Ver. Deut. Ing. 77 (1933) 318–320.
[32] Yu-Heng Tseng, Joel H. Ferzinger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2003) 593–623.
[33] T. Ye, R. Mittal, H.S. Udaykumar, W. Shyy, An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries, J. Comput. Phys. 156 (1999) 209–240.