Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 210 (2005) 1–31
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Sharp interface Cartesian grid method I: An easily implemented technique for 3D moving boundary computations

S. Marella, S. Krishnan, H. Liu, H.S. Udaykumar *

Department of Mechanical and Industrial Engineering, University of Iowa, 3026 Seamans Center, Iowa City, IA 52242-1527, United States

Received 1 July 2004; received in revised form 18 March 2005; accepted 24 March 2005
Available online 23 May 2005

Abstract

A Cartesian grid method is developed for the simulation of incompressible flows around stationary and moving three-dimensional immersed boundaries. The embedded boundaries are represented using level-sets and treated in a sharp manner without the use of source terms to represent boundary effects. The narrow-band distance function field in the level-set boundary representation facilitates implementation of the finite-difference flow solver. The resulting algorithm is implemented in a straightforward manner in three-dimensions and retains global second-order accuracy. The accuracy of the finite-difference scheme is established and shown to be comparable to finite-volume schemes that are considerably more difficult to implement. Moving boundaries are handled naturally. The pressure solver is accelerated using an algebraic multigrid technique adapted to be effective in the presence of moving embedded boundaries. Benchmarking of the method is performed against available numerical as well as experimental results.

Keywords: Moving boundaries; Sharp interface method; Level-sets; Cartesian grid

1. Introduction

Cartesian grid methods now come in several flavors, both in the diffuse and sharp interface categories. Typically, diffuse interface methods have been deemed to be simpler to implement and therefore have found wide usage [1]. In such methods, the onus is on designing the source terms so that the formulation approaches the sharp interface limit for vanishing interface thicknesses. Although straightforward to

* Corresponding author. Tel.: +1 319 384 0832; fax: +1 319 335 5669.
E-mail addresses: ush@engineering.uiowa.edu, ush@icaen.uiowa.edu (H.S. Udaykumar).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2005.03.031

formulate, sharp interface methods have been considered to be somewhat more difficult to implement and therefore their use has been restricted to some specific situations and to a few practitioners. The present paper and the companion papers [2,3] seek to demonstrate that with a proper framework sharp interface methods can be developed and implemented with considerable ease and applied to a variety of moving boundary problems. It is also shown that going from two to three-dimensions can also be accomplished fairly easily with the framework presented in this paper. A significant point to note is that the current framework advances a Cartesian grid method that applies to a whole variety of physical problems, including multiphase flows [2] and phase change [3].

Cartesian grid methods can be broadly classified in two categories:

(a) *Methods where the interface effects are transmitted through forcing functions:* Perhaps the most commonly used fixed grid approach for moving boundary computations involving solid–fluid as well as fluid–fluid interfaces is the immersed boundary method due to Peskin [4]. In the original immersed boundary method as well as later versions, the interaction of the boundary with the fluid is effected using smoothed δ-functions located at the boundaries. The effect of the boundaries, in particular boundary forces (such as elastic forces in structures, tethering forces and surface tensions) are transmitted to the momentum equations as source (or forcing) terms

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \vec{\nabla}\vec{u} = -\frac{1}{\rho}\vec{\nabla}p + \upsilon\nabla^2\vec{u} + \vec{f}. \tag{1}$$

Here, $\vec{f}$ is the forcing function that represents the effect of the immersed boundary. Typically a numerical δ-function with a support of a few mesh spacings is used to convert singular surface forces (such as surface tension) into volume forces $\vec{f}$. The strategy of transmitting interface effects to the flow field through source terms has also been used to solve problems in multiphase flows [5–9] and solidification [10]. However, one shortcoming of these methods is that discontinuities at the immersed boundary are smeared across a few cell widths. It has been shown that such smearing can adversely impact the accuracy of solutions when the boundary motion is closely coupled with the evolution of surrounding fluid flow [11]. There are also issues involved with stability and stiffness of the computations [12,13], particularly, when the embedded objects deform along with the flow. Improvements to the δ-function based immersed boundary methods have appeared in the literature in recent years [14–16]. In the finite-element setting, the fictitious domain method [17–19] and the immersed finite element method [20] have followed this idea of transmitting the boundary forces to the fluid using an interaction source term. However, a considerable number of papers on immersed boundary methods in recent years have deviated from the use of δ-functions in transmitting boundary forces to the fluid. For example, in the finite difference/finite volume methods presented in several recent papers [21–24], the idea of using a forcing term in the momentum equation has been retained. These (non-smoothed) forces are placed at points that adjoin the immersed boundary (either inside or outside the solid object) in order to impose the appropriate velocity boundary conditions on the solid surface. Thus, unlike the original immersed boundary method of Peskin and its derivatives, these new immersed boundary methods are in fact sharp interface methods.

(b) *Methods where the interface effects are included in the discrete spatial operators:* There are sharp interface methods, however, that do not use forcing terms but incorporate the presence of the embedded boundaries into the *discrete* form of the Navier–Stokes equations. Thus, in contrast with the above approach, the momentum equation is retained in the form

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \vec{\nabla}\vec{u} = -\frac{1}{\rho}\vec{\nabla}p + \upsilon\nabla^2\vec{u}. \tag{2}$$

During the discretization procedure, however, the spatial differential operators $(\vec{\nabla}, \nabla^2)$ in the equation are constructed at points that adjoin the interface in such a way that the interfacial jump conditions

are incorporated. Examples of this class are the immersed interface method [25–27], the sharp interface method [28–30], the ghost fluid method [31,32] and the XFEM method [33–36]. The immersed interface method (IIM) [27] enables a sharp interface treatment by casting the governing equations in a coordinate system with axes oriented with the local normal and tangent to the interface. Problems involving embedded fluid–fluid interfaces with singular sources and jumps in material property across the interface have been solved using this approach [26,27]. The method seeks to preserve second-order accuracy at points adjacent to the interface as well as away from it. In Ghost Fluid Method (GFM) [31], the governing equations are discretized along the Cartesian coordinate directions. Only first-order accuracy is demanded at interface-adjacent points. In both IIM and GFM, jumps and singular sources at the interface are incorporated into the discrete operators in the transport equations. For problems involving the Poisson equations, such as the Stefan problem sharp interface methods with embedded level-sets along the lines of the finite-difference ghost-fluid approach have also been developed by Chen et al. [37] and Gibou et al. [38–40]. The paper by Gibou et al. [40] in particular provides a method for fourth order accurate discretization of the Poisson equation in the ghost-fluid framework. For solid boundaries immersed in fluids, the ghost-fluid approach has been applied in several interesting 3D settings by Yokoi et al. [41,42]. In the sharp interface method [29,43], a finite volume technique is used to discretize the equations within the domains separated by the embedded boundary in such a way that information is not smeared at the immersed boundary. This requires reshaping of the control volumes through which the interface passes and the integration of the weak form of the governing equations over non-rectangular control volumes. Second-order accuracy is maintained at bulk as well as interface-adjacent grid points. Problems involving fluid–structure interactions [29,30] and solidification [7,28,44,45] have been solved using this approach. Other approaches for finite-volume computation of flows around embedded boundaries involve employing volume-fractions in the mixed cells, such as in [46,47]. In the finite element community, the XFEM method [48] follows a similar strategy, in that the elements through which the boundary passes are enriched (i.e., these elements are subdivided and conform locally to the immersed boundary; stated differently degrees of freedom are added) to facilitate integration of the weak form of the governing equations.

2. The current sharp-interface method

The critical issues that arise in developing sharp interface Cartesian grid methods for three-dimensional moving boundaries problems are:

(a) *Representation of embedded interfaces:* Explicit surface tracking (involving surface meshing and remeshing) can be challenging for complex 3D moving boundaries, particularly in the presence of sharp edges, cusps, instabilities and topological changes in the boundaries. Geometric details such as intersections between the triangulated surface mesh and the underlying flow solver mesh need to be computed repeatedly. Normal and curvature computations need to be performed accurately on the surface. The latter quantity in particular is difficult to compute accurately on a surface that is defined using piecewise surface patches [49]. The level-set technique presents a solution to many of these problems due to the implicit interface representation and built-in regularization due to the entropy-satisfying solutions obtained from level-set advection. In this approach, at each mesh point in a narrow band surrounding the interface the signed normal distance to the interface is stored. The interface is implicitly contained in this information as the zero-level surface and can be deduced to desired accuracy. In the case where a solid geometry is represented using a surface mesh it is relatively straightforward to construct a level-set field from the given geometry using fast marching [50] or other methods [51]. Representation of all interfaces (whether solid–fluid or fluid–fluid) using level-sets simplifies discretization at computational points adjacent to the interfaces as shown later in this paper. In the present context, a significant advantage of the level-set representation is

that it is possible to maintain a sharp-interface treatment of all boundaries. This fact underlies, for instance, the Ghost Fluid Method (GFM) for fluid–fluid boundaries presented in [32].

(b) *Solving for flows around the immersed boundaries:* Since interfaces cut through the mesh in arbitrary fashion finite volume discretization requires reshaping of the cells cut by the interfaces. Numerous configurations of the cut-cells can arise in 3D and special treatment for each case is tedious, from an implementation as well as bookkeeping standpoint. This can be avoided by employing a finite-difference discretization of the strong form of the governing equations. In adopting the finite-difference approach, the main concern is the accuracy and conservation properties of the flow solver. Fortunately, for the Cartesian grid approach, the deviation of the finite-difference approach from the finite-volume approach appears only at the grid points adjoining the interface. It is necessary, however, to evaluate the global accuracy of the solutions obtained from the finite-difference and finite-volume discretizations for benchmarked flow fields. Here, a second-order accurate Cartesian grid based finite-difference scheme is used to discretize the incompressible Navier–Stokes equations. The discretization depends essentially on convolving the differential operators with the distance function field, an idea applied to the solution of Poisson solvers in the work of Gibou et al. [38,39]. The result is an easily implemented algorithm where the discretization of the governing equations at all points (i.e., away from as well as adjoining the interface) can be presented in a unified format. Thus, the present algorithm handles embedded solid–fluid interfaces (using the sharp-interface method detailed in this paper) and fluid–fluid interfaces (using the Ghost Fluid method, as demonstrated in Part II [2]) in a unified fashion. The present algorithm, with only minor adaptations, has been applied to problems such as droplet-wall interaction [2] and solidification [3].

3. Discretization of governing equations

3.1. Equations to be solved

The governing equations for incompressible flow, with variations in density due to temperature or solutal inhomogeneities are (under Boussinesq approximation):

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

$$\frac{\partial\vec{u}}{\partial t}+\vec{u}\cdot\vec{\nabla}\vec{u}=-\frac{1}{\rho_0}\vec{\nabla}p'+\frac{\mu}{\rho_0}\nabla^2\vec{u}+\frac{\rho'}{\rho_0}\vec{g}, \tag{4}$$

$$\frac{\mathrm{D}\rho'}{\mathrm{D}t}=-w\frac{\partial\rho_{\mathrm{B}}}{\partial z}, \tag{5}$$

where ρ' and p' are the perturbed components of density and pressure defined as $\rho'=\rho-\rho_{\mathrm{B}}$, $p'=p-p_{\mathrm{B}}$.

In the above set of equations, $\vec{u}$ is the velocity field and, ρ_0, μ and $\vec{g}$ ($=-g\hat{k}$, $\hat{k}$ is the unit vector in z-direction) correspond to the reference density, viscosity and acceleration due to gravity, respectively. The basic undisturbed distributions of density and pressure (ρ_{B}, p_{B}) satisfy hydrostatic equilibrium ($\mathrm{d}p_{\mathrm{B}}/\mathrm{d}z=-\rho_{\mathrm{B}}g$). The characteristic velocity, length and density scales chosen for non-dimensionalizing the governing equations are the inlet flow velocity U_0, characteristic length of the immersed object D and the reference density ρ_0, respectively. The governing equations transform to the following dimensionless form:

$$\vec{\nabla}\cdot\vec{u}=0, \tag{6}$$

$$\frac{\partial\vec{u}}{\partial t}+\vec{u}\cdot\vec{\nabla}\vec{u}=-\vec{\nabla}p+\frac{1}{Re}\nabla^2\vec{u}+\frac{\rho}{Fr^2}\vec{g}, \tag{7}$$

$$\frac{\partial\rho}{\partial t}+\vec{u}\cdot\vec{\nabla}\rho=w. \tag{8}$$

In the above equations, $Re = \rho_0 U_0 D/\mu$ and $Fr = U_0/ND$ represent the Reynolds number and internal Froude number, respectively, where N is the buoyancy frequency defined as $N = \sqrt{(g/\rho_0)(\mathrm{d}\rho_B/\mathrm{d}z)}$. In the computations performed the density variations in the domain are neglected in all cases except those involving stratified flows [52].

The scalar (temperature, species concentration) transport equations take the general form:

$$\beta_t \frac{\partial \xi}{\partial t} + \vec{\nabla} \cdot \vec{u}\xi = \beta_d \nabla^2 \xi, \tag{9}$$

where β_t and β_d are material constants associated with the time-dependent and diffusive terms, respectively.

3.2. Flow solver

A cell-centered collocated arrangement of the flow variables is used to discretize the governing equations. A two-step fractional step method [43,53] is used to advance the solution in time. The first step evaluates an intermediate velocity by solving an unsteady advection–diffusion equation

$$\frac{\vec{u}^* - \vec{u}^n}{\Delta t} = -\vec{u} \cdot \vec{\nabla}\vec{u} + \frac{1}{Re}\nabla^2 \vec{u} + \frac{\rho}{Fr^2}\vec{g}, \tag{10}$$

where the intermediate velocity $\vec{u}^*$ is evaluated with central-difference discretization schemes for convection and diffusion terms. The convective terms are treated explicitly and discretized using a second-order accurate Adams–Bashforth method

$$\vec{u} \cdot \vec{\nabla}\vec{u} = \tfrac{1}{2}(3\vec{u}^n \cdot \vec{\nabla}\vec{u}^n - \vec{u}^{n-1} \cdot \vec{\nabla}\vec{u}^{n-1}). \tag{11}$$

The diffusion terms are treated semi-implicitly using Crank–Nicholson scheme

$$\frac{1}{Re}\nabla^2 \vec{u} = \frac{1}{2Re}(\nabla^2 \vec{u}^* + \nabla^2 \vec{u}^n). \tag{12}$$

The second fractional-step involves the correction of the intermediate velocity field $\vec{u}^*$ to enforce mass conservation

$$\frac{\vec{u}^{n+1} - \vec{u}^*}{\Delta t} = -\vec{\nabla} p, \tag{13}$$

where the pressure field p is evaluated to impose a divergence-free velocity field at time step $n + 1$. This is done by taking the divergence of Eq. (13) to obtain a Poisson equation for pressure

$$\nabla^2 p = \frac{\vec{\nabla} \cdot \vec{u}^*}{\Delta t}. \tag{14}$$

The final semi-discrete form of the equations including each of the above discretization schemes is as follows:

$$\frac{\vec{u}^*}{\Delta t} - \frac{1}{2Re}\nabla^2 \vec{u}^* = \frac{\vec{u}^n}{\Delta t} + \frac{\rho \vec{g}}{Fr^2} + \frac{1}{2Re}\nabla^2 \vec{u}^n - \frac{1}{2}(3\vec{u}^n \cdot \vec{\nabla}\vec{u}^n - \vec{u}^{n-1} \cdot \vec{\nabla}\vec{u}^{n-1}), \tag{15}$$

$$\nabla^2 p^{n+1} = \frac{\vec{\nabla} \cdot \vec{u}^*}{\Delta t}. \tag{16}$$

The intermediate velocity is then corrected to obtain the final divergence-free velocity field

$$\vec{u}^{n+1} = \vec{u}^* - \Delta t \vec{\nabla} p. \tag{17}$$

The advection–diffusion equations for scalar (heat and species) transport are discretized in a similar manner, i.e.

$$\beta_t \frac{\xi^{n+1}}{\Delta t} - \frac{\beta_d}{2}\nabla^2 \xi^{n+1} = \beta_t \frac{\xi^n}{\Delta t} + \frac{\beta_d}{2}\nabla^2 \xi^n - \frac{1}{2}(3\vec{u}^n \cdot \vec{\nabla}\xi^n - \vec{u}^{n-1} \cdot \vec{\nabla}\xi^{n-1}). \tag{18}$$

3.3. Implicit interface representation using levelsets

Embedded surfaces are represented implicitly on the mesh using a standard level-set approach [54–56]. In addition to the flow variables, the level-set method advects a scalar field ϕ_l, where subscript l denotes the lth embedded interface. The value of ϕ_l at any point is the signed normal distance from the lth interface with $\phi_l < 0$ inside the immersed boundaries and $\phi_l > 0$ outside. The interface location is implicitly embedded in the ϕ_l-field since the $\phi_l = 0$ contours represent the lth immersed boundary.

In case of moving interfaces, the motion of the boundary is tracked by advecting the level-set using

$$(\phi_l)_t + \vec{V}_l \cdot \vec{\nabla}\phi_l = 0, \tag{19}$$

where $\vec{V}_l$ is the lth level-set velocity field. A fourth-order ENO scheme in space and fourth-order Runge–Kutta integration in time are used for the evolution of the level-set field. Since $\vec{V}_l$ is prescribed by the physics only on the interface (i.e., on the zero-level-set), the value of velocity at the grid points that lie in the narrow band around the zero-level set needs to be obtained. This is done by extension of the interfacial velocity [54,57] away from the front using

$$\psi_\tau + \vec{V}_{ext} \cdot \vec{\nabla}\psi = 0, \tag{20}$$

where ψ is any quantity (such as interface velocity components $(\vec{V}_l)_x$ or $(\vec{V}_l)_y$) that needs to be extended away from the interface. A choice for the extension velocity is $\vec{V}_{ext} = \text{sign}(\phi_l)\frac{\nabla\phi_l}{|\nabla\phi_l|}$. This populates the narrow band around each interface in the time $\tau = O(\Delta x)$ with a level-set velocity that has been extended outward from the interface in a direction normal to it. A reinitialization procedure [58,59] is carried out after level-set advection to return the ϕ_l-field to a signed distance function, i.e., to satisfy $|\vec{\nabla}\phi_l| = 1$. Suppose $(\phi_l)_0$ is the level-set field prior to re-initialization. The following equation is solved to steady-state to re-initialize the level-set field:

$$(\phi_l)_\tau + \vec{w} \cdot \nabla\phi_l = \text{sign}(\phi_l), \tag{21}$$

$$\vec{w} = \text{sign}((\phi_l)_0)\frac{\nabla(\phi_l)_0}{|\nabla(\phi_l)_0|}, \quad \text{where } \text{sign}((\phi_l)_0) = \frac{(\phi_l)_0}{\sqrt{(\phi_l)_0^2 + (\Delta x)^2}} \tag{22}$$

with the initial condition $\phi_l(\vec{x}, 0) = (\phi_l)_0(\vec{x})$.

The calculation of normal and curvature of the interface from the level-set field is simple. The normal is given by

$$\vec{n} = \vec{\nabla}\phi_l / |\vec{\nabla}\phi_l| \tag{23}$$

and curvature is obtained from

$$\kappa = -\vec{\nabla} \cdot \vec{n}. \tag{24}$$

3.4. Discretization of operators

3.4.1. Classification of grid points

The grid points on the Cartesian mesh are classified as *bulk points* and *interfacial points*. The latter are points that lie immediately adjacent to the immersed interface. Fig. 1(a) illustrates an immersed interface and the points that are classified as *interfacial points*, i.e., points that satisfy the condition

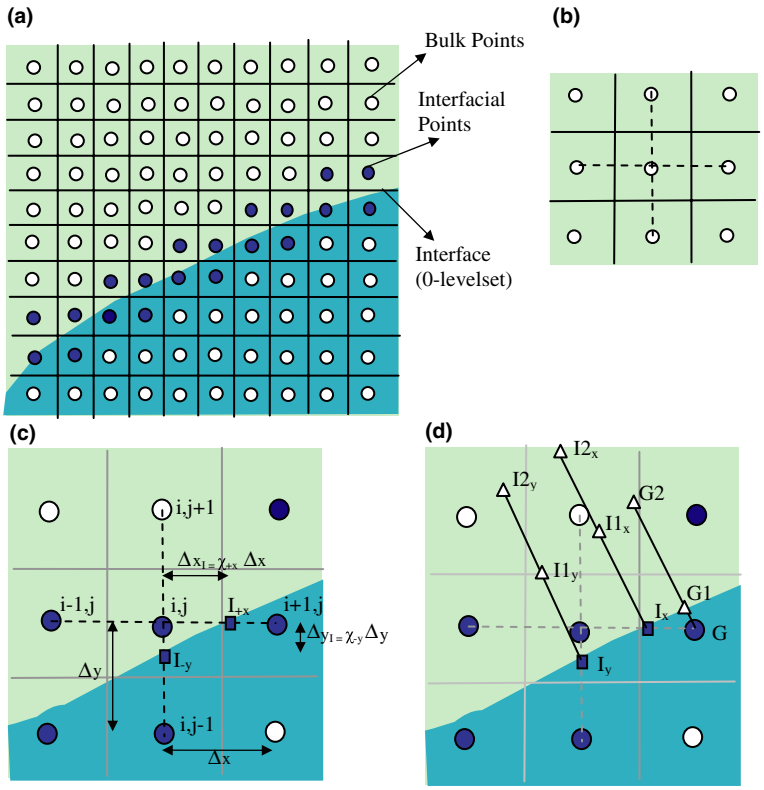

Fig. 1. (a) Definition of the bulk (clear circles) and interfacial (filled circles) points. The interface is given by the 0-levelset. (b) Standard 5-point bulk point stencil in two-dimensions. (c) The configuration of a typical interfacial point. (d) System for evaluating the Neumann boundary condition on the interface and evaluation of ghost pressures.

$(\phi_I)_{i,\ j}(\phi_I)_{\mathrm{nb}} \leqslant 0$, where subscript nb denotes an immediate neighbor along the coordinate directions. The discrete operators at the interfacial points are different from those that apply at the bulk points. The strategy adopted to deal with this situation for the differential operators in the governing equation are discussed below.

3.4.2. Discretization at bulk points

A standard 5-point central-difference stencil applies for a typical *bulk point* (i.e., a grid point that does not adjoin the embedded interface) in a two-dimensional Cartesian mesh. While only two-dimensional situations are shown in the figures for ease of visualization, the discussion below carries over to three-dimensions. The three-dimensional counterpart would involve a 7-point stencil and the discretization for the momentum equations is identical in all the three-dimensions as described below.

The second derivative w.r.t. x in the diffusion term is discretized as follows:

$$\frac{\partial^2 \psi}{\partial x^2} = \frac{\psi_{i+1,j} - \psi_{i,j}}{\Delta x^2} - \frac{\psi_{i,j} - \psi_{i-1,j}}{\Delta x^2}, \tag{25}$$

while the convection term in the x-direction is obtained from

$$\frac{\partial u\psi}{\partial x} = \frac{u_{i+1/2,j}\psi_{i+1/2,j} - u_{i-1/2,j}\psi_{i-1/2,j}}{\Delta x}, \tag{26}$$

where u is the x-component of the velocity and

$$\psi_{i\pm1/2,j} = \frac{\psi_{i\pm1,j} + \psi_{i,j}}{2}, \tag{27}$$

where $u_{i\pm1/2,j}$ are the velocities on the cell faces. Similar considerations apply along the y- and z-directions.

3.4.3. Discretization at interfacial points

As pointed out before, the main challenge in sharp interface fixed-grid methods is to accurately impose interfacial conditions in the discrete system of equations. Moreover, a sharp-interface method demands one-sided discretization for all the partial derivatives to avoid smearing of the interfaces. For the case of immersed solid–fluid boundaries, as in fluid–structure interaction problems, the no-slip and no-penetration velocity boundary conditions are applied on the solid surfaces. For small Strouhal numbers, the Neumann condition for pressure applies on such boundaries [45]. These boundary conditions are then supplied to the governing equations through the discretization at the interfacial points as described below.

(a) *$\partial^2\psi/\partial x^2$ with a Dirichlet boundary condition on the embedded boundary:* Second derivatives need to be computed in the diffusion terms in the momentum as well as scalar transport equations and in these cases typically a Dirichlet condition applies at the boundary. With particular reference to the point (i,j) in Fig. 1(c), this point lies in the fluid and the velocities, scalars (temperature and species concentration) and pressure are computed there. Note that for points that lie in the solid only the temperature and species fields are computed. In discretizing $\partial^2\psi/\partial x^2$ at point (i,j) in Fig. 1(c), the neighbor point $(i+1,j)$ lies across the interface in the solid and hence cannot be used in discretization. In order to include the interfacial values (i.e., apply interfacial conditions), it is necessary to find the location where the 0-level-set intersects the line joining the cell centers (indicated by the square symbols in Fig. 1(c)). The discretization for such operators can be advantageously constructed in a simple manner using the approaches given in [38]. Note that higher-order approaches are also possible as in [40]. However, we will restrict ourselves in the present case to attempting a second-order accurate treatment. Following the schemes in [38], in the following expressions for the coefficients frequent use will be made of the quantity $\chi = \frac{\Delta x_I}{\Delta x}$ (see Fig. 1(c)), where Δx_I is the distance between the cell center and the intersection of cell centerlines with the interface (filled square); Δx is the nominal cell width. By noting that the intersection point has a zero level-set value, χ can be easily evaluated using the level-set information at (i,j) and $(i+1,j)$

$$\chi = \frac{\Delta x_I}{\Delta x} \cong \left|\frac{(\phi_l)_{I_x} - (\phi_l)_{i,j}}{(\phi_l)_{i+1,j} - (\phi_l)_{i,j}}\right| = \left|\frac{0 - (\phi_l)_{i,j}}{(\phi_l)_{i+1,j} - (\phi_l)_{i,j}}\right|. \tag{28}$$

Note that in evaluating the quantity χ a linear profile is assumed for the distance function between adjacent grid points. Higher-order approximations of the geometry can be implemented as well [60,61].

The second-derivative can be estimated to second-order accuracy using the form

$$\frac{\partial^2\psi}{\partial x^2} = \alpha_I\psi_I + \alpha_{i,j}\psi_{i,j} + \alpha_{i-1,j}\psi_{i-1,j} + \alpha_{i-2,j}\psi_{i-2,j}, \tag{29}$$

where ψ_I is the value on the interface at the location I_{+x} (Fig. 1(c)).

Using Taylor series expansions for each of ψ_I, $\psi_{i-1,j}$, $\psi_{i-2,j}$ about point (i,j) and further demanding that $\partial^2\psi/\partial x^2$ be estimated to $O(\Delta x^2)$ yields the following expressions for the coefficients:

$$\alpha_I = 6/(\chi(\chi+1)(\chi+2)\Delta x^2), \tag{30a}$$
$$\alpha_{i-1,j} = (4-2\chi)/((\chi+1)\Delta x^2), \tag{30b}$$
$$\alpha_{i-2,j} = (\chi-1)/((\chi+2)\Delta x^2), \tag{30c}$$
$$\alpha_{i,j} = -\alpha_I - \alpha_{i-1,j} - \alpha_{i-2,j}. \tag{30d}$$

In practice, the above implementation may present difficulties due to the singular behavior of α_I as $\chi \to 0$. Therefore, for small values of $\chi(<0.01)$, i.e., when the interface is very close to the mesh point, the value χ is replaced by $\max(\chi,0.01)$. This involves a slight perturbation of the boundary within a grid cell and decreases the order of accuracy locally from second-order. However, this situation arises at only a few mesh points and the global accuracy is not impacted. Note that a first-order approximation is given by:

$$\alpha_I = 2/(\chi(1+\chi)\Delta x^2), \tag{31a}$$
$$\alpha_{i-1,j} = 2/((\chi+1)\Delta x^2), \tag{31b}$$
$$\alpha_{i,j} = -\alpha_I - \alpha_{i-1,j}, \tag{31c}$$
$$\frac{\partial^2\psi}{\partial x^2} = \frac{2}{\chi(1+\chi)}\frac{(\psi_I - \psi_{i,j})}{\Delta x^2} - \frac{2}{(1+\chi)}\frac{(\psi_{i,j} - \psi_{i-1,j})}{\Delta x^2}. \tag{31d}$$

The singularity with respect to χ remains in this case as well. However, positivity of the off-diagonal coefficients is maintained in the first-order case while the second-order form will lead to a negative coefficient $\alpha_{i-2,j}$. In practice this negatively impacts the convergence of the iterative solver used for solving the discrete system of equations; to maintain robustness a first-order treatment for the diffusion term is employed at the interfacial points in the present calculations. While this choice lowers the order of approximation in the lower-dimensional set of interfacial cells, global second-order accuracy is still maintained as shown in the results.

(b) *Convection and divergence terms ($\partial(u\psi)/\partial x$ and $\partial\psi/\partial x$) with Dirichlet conditions on the boundary:* As for the second-derivative terms above, the discretization scheme for $\partial(u\psi)/\partial x$ consists of contributions from points in the same phase. Again, following the approach taken by Gibou et al. [38] for the Poisson equation, the differential operator for the convection term is obtained in the following form:

$$\frac{\partial u\psi}{\partial x} = \lambda_I(u\psi)_I + \lambda_{i-1/2,j}(u\psi)_{i-1/2,j} + \lambda_{i-3/2,j}(u\psi)_{i-3/2,j}. \tag{32}$$

Note that, as in [29], to avoid pressure–velocity decoupling in the current collocated variable arrangement cell-face velocities are also stored along with cell-center velocities [53]. These cell-face velocities are used in evaluating the convective fluxes in Eq. (32). By employing Taylor expansions for each of the $\psi_{i',j'}$ in Eq. (32) about point (i,j,k) and demanding an $O(\Delta x^2)$ scheme the following expressions are obtained for the constants:

$$\lambda_{i-1/2,j} = (2\chi-3)/((2\chi+1)\Delta x), \tag{33a}$$
$$\lambda_{i-3/2,j} = (1-2\chi)/((3+2\chi)\Delta x), \tag{33b}$$
$$\lambda_I = 8/((4\chi^2+8\chi+3)\Delta x). \tag{33c}$$

The general form and the coefficients for a first-order scheme in this case are:

$$\frac{\partial u\psi}{\partial x} = \lambda_I(u\psi)_I + \lambda_{i-1/2,j}(u\psi)_{i-1/2,j}, \tag{34a}$$
$$\lambda_{i-1/2,j} = -2/((1+2\chi)\Delta x), \tag{34b}$$
$$\lambda_I = 2/((1+2\chi)\Delta x). \tag{34c}$$

The above general form can be rewritten as follows:

$$\frac{\partial u\psi}{\partial x} = \frac{2}{(1+2\chi)} \frac{((u\psi)_I - (u\psi)_{i,j})}{\Delta x}. \tag{35}$$

Since the convection terms are explicitly computed the second-order approximation can be employed except at those points where two opposing interfaces approach to within a mesh point. In this exigency, the first-order approximation needs to be adopted.

(c) $\partial^2\psi/\partial x^2$ *operator with a Neumann boundary condition on the embedded boundary:* This situation arises in the pressure Poisson equation for points that adjoin the embedded boundary. The stability and accuracy of the flow solver depends critically on the construction of this term. In fact, devising a discrete form for the Laplace operator with a Neumann condition on the immersed interface proved to be the key to the robustness of the overall flow solver.

From Fig. 1(c), it is evident that when discretizing the above operator the boundary condition that will apply at point I_{+x} is $\partial\psi/\partial n = 0$. It is not immediately clear how such a Neumann condition can be incorporated into the discrete form of the above operator. Apart from being crucial to the robustness of the overall method, the treatment of the pressure boundary condition is a key distinguishing feature of the finite-difference approach adopted here as opposed to the finite-volume approach detailed in previous work [29]. In the latter a weak form of the pressure Poisson equation was employed, i.e.

$$\oint \frac{\partial p}{\partial n}\, \mathrm{d}S = \oint \frac{\vec{u}^* \cdot \vec{n}}{\Delta t}\, \mathrm{d}S. \tag{36}$$

The interfacial cells were reshaped into irregular shaped cells where one of the cell edges coincided with the interface. Due to the weak form above the Neumann boundary condition for pressure is easily incorporated by setting the interfacial contribution to zero (i.e., $\partial p/\partial n = 0$) implicitly in the discrete pressure Poisson equation. However, since the strong form is employed in the present finite-difference scheme on a Cartesian grid, one has to impose a Neumann boundary condition on the pressure in some way in the discrete Laplace operator. Several approaches were tried for discretizing terms such as $\partial^2\psi/\partial x^2$ with a Neumann condition on the immersed boundary and the one chosen for robustness is described below.

The Laplace operator in the pressure Poisson equation is assembled as for the Dirichlet boundary condition case (i.e., Eq. (29)). The interfacial pressure is found using a normal probe approach [7,62] to satisfy the Neumann condition. Looking at Fig. 1(d), the interface pressure can be estimated by extending a normal from point I and placing two points distant Δx apart along the normal. The locations of the points I_x, $I1_x$ and $I2_x$ on the probe are therefore:

$$\vec{x}_{I_x} = \vec{x}_{i,j} + \Delta x_I \vec{i}, \tag{37a}$$

$$\vec{x}_{I1_x} = \vec{x}_{I_x} + \vec{N}_{I_x} \Delta x, \tag{37b}$$

$$\vec{x}_{I2_x} = \vec{x}_{I_x} + 2\vec{N}_{I_x} \Delta x. \tag{37c}$$

Similarly:

$$\vec{x}_{I_y} = \vec{x}_{i,j} + \Delta y_I \vec{j}, \tag{38a}$$

$$\vec{x}_{I1_y} = \vec{x}_{I_y} + \vec{N}_{I_y} \Delta x, \tag{38b}$$

$$\vec{x}_{I2_y} = \vec{x}_{I_y} + 2\vec{N}_{I_y} \Delta x. \tag{38c}$$

Fitting a quadratic to the pressure field along the normal erected and demanding that $\frac{\partial\psi}{\partial n} = 0$ at point I, one obtains

$$\psi_{I_{x/y}} = \tfrac{4}{3}\psi_{I1_{x/y}} - \tfrac{1}{3}\psi_{I2_{x/y}}, \tag{39}$$

In the above expressions the normal vector appears in several places. The normal at any point is easily obtained by bilinear interpolation from the values at the grid points. Note that the values of ψ at the points $I1_{x/y}$ and $I2_{x/y}$ are required in the above equation. These are calculated by bilinear interpolation from the surrounding mesh points. One issue that arises while evaluating the $\psi_{I1x/y}$ and $\psi_{I2x/y}$ is that for certain interface orientations the bilinear interpolation may involve a point that lies in the solid. Thus, a single layer of ghost values of pressure is computed and stored at the interfacial points in the solid. The ghost values of pressure are also obtained with the condition that the Neumann condition applies at the interface. Thus, a normal to the interface is erected from the ghost point as shown in Fig. 1(d), where the locations of the points on the normal are:

$$\vec{x}_{G1} = \vec{x}_G + \vec{N}_G|\phi_G|, \tag{40a}$$

$$\vec{x}_{G2} = \vec{x}_{G1} + \vec{N}_G\Delta x. \tag{40b}$$

The normal at point G is obtained from the level-set field using Eq. (23). Fitting a quadratic to the pressure field along the normal and demanding that $\frac{\partial\psi}{\partial n} = 0$ be satisfied at the point $G1$ (i.e., on the solid boundary) leads to

$$\psi_G = \frac{\psi_{G1}d_2^2 - 2\psi_{G1}d_1d_2 + \psi_{G2}d_1^2}{d_2^2 - 2d_1d_2 + d_1^2}, \tag{41}$$

where

$$d_1 = |(\phi_I)_G| \quad \text{and} \quad d_2 = |(\phi_I)_G| + \Delta x. \tag{42}$$

The interfacial value of the pressure ψ_{G1} is obtained from Eq. (39) above and the value at $G2$ is obtained by bilinear interpolation. Note that the value of the ghost pressure and the interfacial pressure become interdependent through Eqs. (39) and (41). The determination of the interfacial pressure and ghost pressure are embedded within an iterative solver for the pressure and therefore the ghost and interfacial value of pressure converge along with the pressure field at the fluid computational points. This second scheme works reliably for the entire range of Reynolds numbers and flow problems tested using the present solver. The above scheme applies in three-dimensions as well.

3.4.4. A general form of the discretization for the operators

The interface can cut through the mesh in an arbitrary fashion leading to several different configurations of interfacial points. Some situations are illustrated in Fig. 2. The central idea here is that the above discretization procedures can be generalized in a straightforward way (in 3D) for implementation purposes without concerning oneself with details of the embedded geometry. Each of the spatial differential operators that appear in Eqs. (10)–(18) can be discretized by considering derivatives along the coordinate directions and the final 3D discrete form assembled in a unified manner by only the signed distance function on the mesh. The following expressions apply where multiple (say $L_{\max}$) embedded boundaries are present in the flow. For components along the x-direction the scheme is as follows.

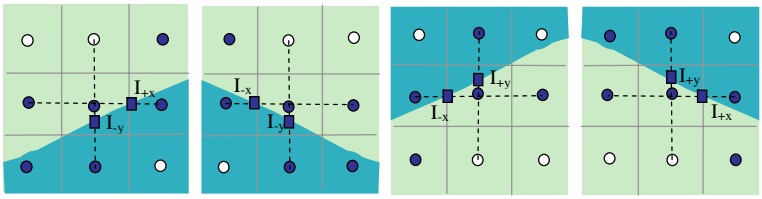

Fig. 2. Some of the possible interfacial point situations in the two-dimensional case.

Define a sign function:

$$(s_l)_{\pm x} = \left\{ \frac{(\phi_l)_{i,j}(\phi_l)_{i\pm1,j}}{|(\phi_l)_{i,j}(\phi_l)_{i\pm1,j}|} \right\},$$
$$s_{\pm x} = \min_{l=1,L_{\max}} \{(s_l)_{\pm x}\}. \tag{43a}$$

The switching function:

$$\chi_{\pm x} = \min_{l=1,L_{\max}} \left\{ |\max((s_l)_{\pm x}, 0)| + \frac{|(\phi_l)_{i,j}|}{|(\phi_l)_{i,j}| + |(\phi_l)_{i\pm1,j}|} |\min((s_l)_{\pm x}, 0)| \right\}. \tag{43b}$$

(a) *The Laplacian operator:*
Then the second derivative w.r.t. x is

$$\frac{\partial^2\psi}{\partial x^2} = \alpha_{+x}\frac{(\psi_{+x} - \psi_{i,j})}{\gamma_x \Delta x^2} - \alpha_{-x}\frac{(\psi_{i,j} - \psi_{-x})}{\gamma_x \Delta x^2}. \tag{43c}$$

For the first-order scheme in the interfacial cells, as described above, the coefficients take the form:

$$\psi_{\pm x} = \psi_{I_{\pm x}} |\min(s_{\pm x}, 0)| + \psi_{i\pm1,j} |\max(s_{\pm x}, 0)|, \tag{43d}$$
$$\chi^d_{\pm x} = \chi_{\pm x}, \tag{43e}$$
$$\alpha_{\pm x} = \frac{1}{\chi^d_{\pm x}}, \tag{43f}$$
$$\gamma_x = \frac{(\chi^d_{-x} + \chi^d_{+x})}{2}. \tag{43g}$$

(b) *The advection and divergence terms (of the form* $\frac{\partial u\psi}{\partial x}$*):*
Once again, following the scheme presented in Eqs. (33)–(34) for the interfacial cells one obtains the general form as:

$$\left(\frac{\partial u\psi}{\partial x}\right)_{i,j} = \frac{(u\psi)_{+x} - (u\psi)_{-x}}{\lambda_x \Delta x}, \tag{44a}$$
$$(u\psi)_{\pm x} = (u\psi)_{I_{\pm x}} |\min(s_{\pm x}, 0)| + (u\psi)_{i\pm1/2,j} |\max(s_{\pm x}, 0)|, \tag{44b}$$
$$\chi^c_{\pm x} = \frac{(1 + |\min(s_{\pm x}, 0)|)}{2}\chi_{\pm x}, \tag{44c}$$
$$\lambda_x = \chi^c_{-x} + \chi^c_{+x}. \tag{44d}$$

Note that this scheme works for the situations where the point (i,j) lies in the fluid and either both, one or none of the neighbors $(i+1,j)$ and $(i-1,j)$ lies in the solid phase. Also, in constructing the three-dimensional Laplacian operator the coefficients $\alpha_{\pm y}/(\gamma_y \Delta y^2)$ and $\alpha_{\pm z}/(\gamma_z \Delta z^2)$ can be calculated in a manner identical to that for $\alpha_{\pm x}/(\gamma_x \Delta x^2)$ above. Note that the simplicity of the algorithm results from the fact that unlike in the finite-volume method, here only the interfacial locations $I_{\pm x/y/z}$ (Fig. 2) are of significance for the discretization procedure. These locations are easily obtained in 3D by using the distance function information carried by the level-set field. In fact, it is not necessary to explicitly compute the locations of points $I_{\pm x/y/z}$ since the switching function $\chi_{\pm x/y/z}$ transmits all the necessary geometric information to the coefficient assembly procedure.

Based on the above templates, the discretized forms of the three-dimensional spatial differential operators that appear in Eqs. (10)–(18) are:

$$\nabla^2\psi = \begin{cases} \alpha_{+x}\frac{(\psi_{+x}-\psi_{i,j,k})}{\gamma_x\Delta x^2} - \alpha_{-x}\frac{(\psi_{i,j,k}-\psi_{-x})}{\gamma_x\Delta x^2} + \alpha_{+y}\frac{(\psi_{+y}-\psi_{i,j,k})}{\gamma_y\Delta y^2}, \\ -\alpha_{-y}\frac{(\psi_{i,j,k}-\psi_{-y})}{\gamma_y\Delta y^2} + \alpha_{+z}\frac{(\psi_{+z}-\psi_{i,j,k})}{\gamma_z\Delta z^2} - \alpha_{-z}\frac{(\psi_{i,j,k}-\psi_{-z})}{\gamma_z\Delta z^2}, \end{cases} \tag{45}$$

$$\nabla\cdot(\vec{u}\psi) = \frac{((u\psi)_{+x}-(u\psi)_{-x})}{\lambda_x\Delta x} + \frac{((v\psi)_{+y}-(v\psi)_{-y})}{\lambda_y\Delta y} + \frac{((w\psi)_{+z}-(w\psi)_{-x})}{\lambda_z\Delta z}. \tag{46}$$

Note that in the advection term above, the cell-face velocities (stored separately from the cell-center velocities) are used in evaluating the fluxes. The convected variable ψ at the cell face is estimated by linear interpolation as in the central-difference scheme. For the pressure Poisson equation, the same discrete operator form as above (Eq. (46)) is used to evaluate the divergence term on the right hand side of Eq. (16), i.e., in that case $\psi = 1$ and $\vec{u}$ is replaced by $\vec{u}^*$.

4. Velocity correction

Once the intermediate velocity and pressure fields have been obtained as described above, the velocity correction step is performed to update to a divergence-free velocity field. For grid points that lie away from the interface this is straightforward. For points that lie next to the immersed boundary the corrections are performed as for the particular case of point (i,j) illustrated in Fig. 1(c), viz.

$$u_{i,j}^{n+1} = u_{i,j}^{*} - \Delta t\frac{(p_{+x}-p_{-x})}{\lambda_x\Delta x}, \tag{47a}$$

where

$$p_{\pm x} = p_{I_{\pm x}}|\min(s_{\pm x},0)| + p_{i\pm1/2,j}|\max(s_{\pm x},0)|, \tag{47b}$$

$$\lambda_x = \chi_{-x}^{c} + \chi_{+x}^{c}. \tag{47c}$$

Note that cell-face velocities are corrected independently

$$\vec{u}_{\text{face}}^{n+1} = \vec{u}_{\text{face}}^{n} - \Delta t(\nabla p^{n+1})_{\text{face}}. \tag{48}$$

The pressure gradient at the cell face is obtained based on straightforward central differences. There is no difference between the correction procedures for the cell-face velocities in bulk cells and in interfacial cells. This is because face velocities in directions in which the interface is present are not included in the discrete forms of the momentum and mass balance equations, as seen from Eq. (46).

5. Moving boundaries

In Eulerian sharp interface methods, when the solid boundary moves across a grid point, the state of the point can change from liquid to solid or vice versa. Different approaches have been employed to handle this situation. In ghost-fluid type methods and immersed boundary methods [21,24,32] or fictitious domain methods [63] flow fields are computed within as well as outside the immersed solid object. Thus, when the boundary crosses over a grid point, changing the state from solid to fluid, the newly emerged fluid point simply takes on the flow field variables that were available at that point in the previous time step. In the sharp interface method [29] as well as immersed interface method [11,27] where the flow is computed separately in each subdomain (fluid and solid) separated by the interface and no ghost flow field exists in the solid, a scheme must be devised to obtain the flow field variables at the newly emerged fluid point. Note that the converse case, i.e., the emergence of a grid point that was in the fluid phase into the solid phase presents no issues since the flow field is not computed in the solid phase.

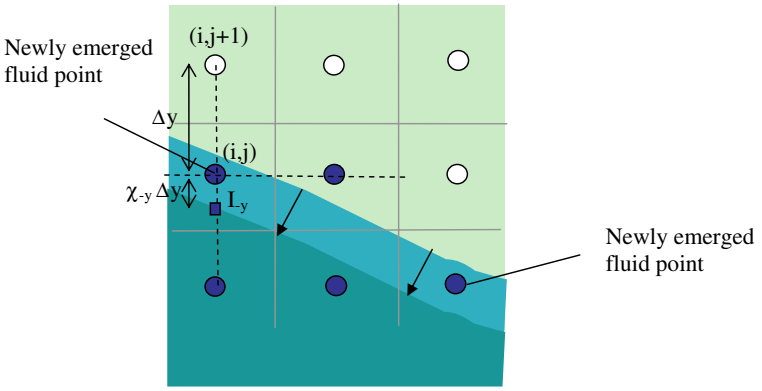

Fig. 3. Illustrations of the emergence of points from the solid to fluid phase when the sharp interface moves through the mesh.

A newly emerged fluid grid point is defined by the condition $(\phi_l)^{n+1}_{i,j}(\phi_l)^n_{i,j} < 0$. Since the point was previously in the solid phase $((\phi_l)^n_{i,j} < 0)$ it had no history in the fluid phase $((\phi_l)^{n+1}_{i,j} > 0)$, i.e., $\vec{u}^n$ (as also ξ^n) does not exist in the fluid phase for such a point. Therefore, these points are to be evolved to time level $n+1$ in a special fashion. Note that since the pressure Poisson equation does not have a time-dependent term the pressure in such a cell can be evaluated as usual once a $\vec{u}*$ value is available after solving the momentum equation. The method to obtain $\vec{u}^n$ (and ξ^n) for such points follows along the lines detailed in [29,30] and is analogous to the approach taken in moving grid formulations when a fresh grid point is inserted following mesh refinement. The value at such points is obtained by interpolation from the known values in the surrounding cells and on the moving boundary (where the boundary conditions are specified). For the particular time step when a grid point changes from solid to fluid phase, the value of $\vec{u}^n$ there is found using a linear interpolation operator spanning points in the fluid and on the interface. The interpolation points that are picked depend on the orientation of the interface in the cell as illustrated in Fig. 2. For the particular case in Fig. 3, the value at the freshly cleared cell (i,j) is calculated as $\psi^n_{i,j} = (\chi_{-y}\psi^n_{i,j+1} + \psi^n_{I_{-y}})/(1+\chi_{-y})$, where χ_{-y} is the distance between the grid point (i,j) and the interfacial point I_{-y}. The interpolation points are chosen depending on the direction of the normal vector at I_{-y} $(\vec{n} = (n_x, n_y))$ and the ratio n_y/n_x. For instance, in the above expression for the case in Fig. 3, points I_{-y} and $(i,j+1)$ are chosen since $(n_y > 0$ and $n_y > n_x)$. Consistent with the difference scheme in the interface-adjacent grid points the treatment at the newly emerged fluid points is first-order accurate. Note that this procedure is equivalent, in analogy with purely Lagrangian (moving grid) methods, to interpolating the value of variables to a newly inserted point after mesh refinement from values at the old mesh points (i.e., before refinement). In diffuse interface Eulerian methods (where interfaces may be captured using VOF, level-set, phase field, etc.), where the interfacial forces are spread over the mesh [9,64] this issue of cross-over does not arise since there is no clear-cut interface location and all properties are taken to vary smoothly over a few mesh points.

6. Multigrid with embedded interfaces

6.1. Algebraic multigrid with moving embedded boundaries

The typical problems to be solved using this method involve embedded solid–fluid and fluid–fluid [2] moving interfaces. A well designed geometric multigrid method has been previously shown to be effective in accelerating the Poisson solver for solid–fluid interface problems [30,43]. Geometric multigrid methods

require information on the manner in which the embedded geometry cuts through the mesh to design an effective grid coarsening strategy and special modifications at each grid level have to be made to account for embedded boundaries. As the complexity of the problems increases upon the inclusion of fluid–fluid and solid–fluid–fluid interfaces with property jumps and surface tension, the complexity of designing a geometric multigrid method also increases. algebraic multigrid (AMG), on the other hand, is virtually a 'black-box' solver. In case of fluid–fluid problems with implementation of Ghost Fluid Method, the formulation is likely to have a coefficient matrix with large coefficient jumps depending on the density difference between the two fluids. Assembly of coefficients on coarser meshes for the geometric multigrid method then becomes a complex process. Moreover, in the case of moving boundary problems, coefficient assembly may need to be done at every time step. In moving interface problems, a local mesh refinement (LMR) algorithm with a tree structure is attractive in order to better resolve the interfaces [65–67]. Such a local mesh refinement would result in the coefficient matrix losing its penta-diagonal structure and becoming sparse and unstructured. In such cases, the AMG approach would be a natural choice to accelerate the Poisson solver, since it relies only on the coefficient information on the finest level of mesh. The geometry information carried by the fine grid coefficients is automatically transmitted to the coarser meshes. With some modification to the standard algebraic multigrid algorithm, an effective grid coarsening and solution strategy has been designed that naturally accounts for the presence of embedded objects, whether they are stationary or moving solid–fluid or fluid–fluid interfaces.

6.2. Local coarsening for moving boundary problems

The main components of the multigrid algorithm include the grids from level 1 (finest) to M (coarsest) denoted by Ω^m where m denotes the level of mesh. Coefficient matrices on each level of mesh m are represented by A^m. The interpolation operator (coarse to fine) is represented by I^m_{m+1} and the restriction operator (fine to coarse) by I^{m+1}_m. The restriction operator matrix is the transpose of the interpolation operator and the coefficient matrix on the next coarser level of mesh is computed by a Galerkin type operation as $A^{m+1} = I^{m+1}_m A^m I^m_{m+1}$. The set-up phase includes dividing the set Ω^m into the coarse and fine sets, defining the interpolation matrix I^m_{m+1} and computing the new operator matrix A^{m+1}.

A significant trade-off for the robustness of AMG is the long setup time required to assemble the coarse meshes and coefficients. In the case of stationary boundaries, the set-up needs to be done only once and can be ignored as a one-time investment of CPU time. But for moving boundary problems, the coefficients on the fine mesh, in the vicinity of the interface are re-assembled at every time step, which means that the coarse meshes also need to be re-assembled at every time step. This is not a very efficient process. To solve this problem, a 'local coarsening' strategy is developed which effectively deals with this obvious disadvantage of AMG as a fast solver. In the Cartesian grid framework, when the interface moves, only the coefficients at grid points close to the interface need to be re-assembled. Therefore, it seems unnecessary to re-coarsen and re-assemble the grids and coefficients in the entire domain at every time step. A strategy has been developed to re-coarsen only the cells within the narrow band of the level set field corresponding to the moving interface. This is done at every level of multigrid. At coarser levels of mesh, the level set information is coarsened and only those mesh points that are likely to have dependencies on the interfacial cells are cleared for re-coarsening. Since the interface generally moves less than a grid spacing during a single time step (due to a CFL-type criterion), this process ensures that all the grid points that are likely to be dependent on the interface points are re-coarsened, while the rest of the domain uses the same coarse grid as in the previous time step. Fig. 4 illustrates this 'local' coarsening process. The domain consists of two objects placed in the flow path shown in the figure by thick lines. The coarsened level set tube is marked by the thin lines. The first object is a solid cylinder of non-dimensional radius 0.1, placed at (0.3, 0.7), the second is a bubble of the same radius placed at (0.7, 0.3). The latter fluid–fluid interface has large coefficient jumps because the density of the liquid is 100 times that of the bubble. Fig. 4(a) shows the coarsened

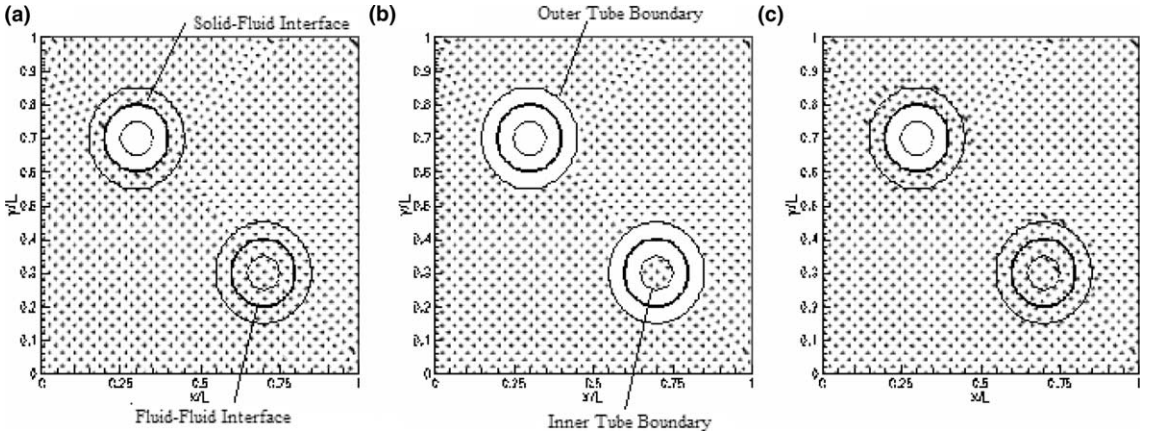

Fig. 4. Local Coarsening process: (a) full coarsening at first time step; (b) outertube cleared of coarse points when the interface moves; (c) outertube recoarsened for multigrid without changing coarse points in non-interface cells.

mesh at the beginning of a time step. At this point, a full coarsening of the domain has been done for the multigrid. Fig. 4(b) shows the starting of coarsening process at the next time step. As shown in the figure, only the mesh points within the level set tube are cleared for coarsening while the remaining domain is maintained as in the previous time step. AMG set up strategy is then applied to the cleared cells. Fig. 4(c) shows the domain that has been re-coarsened locally at the second time step. It can be noticed from Fig. 4(c) that the local coarsening strategy does not necessarily choose the same points that were chosen when the whole domain was coarsened. This implies that with local coarsening, there is a likelihood that the moving interface may leave a trail of irregular coarsening following its path of motion as the solution progresses. From rigorous testing of moving interface problems involving solid–fluid and fluid–fluid interfaces, it was found that the fact that the points chosen by local coarsening are not exactly the same as the ones chosen by full coarsening does not seem to affect the convergence trends of AMG to any serious extent. This is demonstrated later in the results section. Therefore, local coarsening is a strategy that can be used for set up of multigrid for fast solution of the pressure Poisson equation in moving interface problems. If desired, full re-coarsening can be performed periodically at a fixed interval so that all traces of the moving boundary are eliminated periodically in the course of calculation. It has been observed that depending on the size of the interface, the local coarsening strategy takes up only about 10% of the time taken up by full coarsening for most applications.

7. Overall solution procedure

The fractional-step algorithm that includes immersed boundary motion is then as follows:

1. Advance time $t^{n+1} = t^n + \Delta t$.
2. Define the new boundary location by advecting level sets to obtain ϕ^{n+1}. Extend interfacial velocity from the interface into the narrow band.
3. Compute the intermediate velocity fields $\vec{u}^*$:
   (a) Obtain the coefficients for the diffusion term from Eq. (43). Determine interfacial velocities from the extended level-set velocity field by bilinear interpolation.
   (b) Obtain the discrete convection terms from Eq. (44).
   (c) Solve the discrete system of equations using a Line-SOR solver.

4. Compute the scalars ξ_{n+1} (temperature, species concentration, etc.). Follow the same procedure as in 3 above.
5. Compute the pressure field.Conduct a V-cycle multigrid solution of the pressure Poisson equation:
   (1) Assemble finest level coefficients.
      (a) Obtain the interface pressure using Eq. (39).
      (b) Populate ghost pressure values using Eq. (41).
      (c) Obtain the discrete Laplace operator using Eq. (45).
      (d) Calculate the velocity divergence term using Eq. (46).
      (e) Solve finest level.
   (2) Assemble recursively coarse mesh coefficients from the corresponding fine level coefficients and solve.
6. Correct the velocities using Eq. (47) to obtain the final velocities $\vec{u}^{n+1}$.
7. Go to step 1.

8. Results and discussion

8.1. Finite-difference and finite-volume approximations

The accuracy of the current finite-difference and level-set combination (FD + LS) is compared to that of a finite-volume method that used marker tracking (FV + MT) [29]. The main differences between the present FD + LS method and the FV + MT method lie in the manner in which the sharp-interface treatments are constructed in the interface-adjacent grid points. In the FV + MT method, a cut-cell technique was used to reshape the control volumes at such points and the weak form of the governing equations was discretized. However, the present FD + LS scheme is far easier to implement and makes extension to three-dimensions straightforward.

The solutions to scalar diffusion and scalar convection–diffusion problems are chosen as a basis for comparison in this section. The computational setup for all of these cases is a circular cylinder of radius 0.25 placed at the center of a 1×1 domain as shown in Fig. 5(a). For the diffusion problem, Dirichlet boundary conditions are applied at the cylinder surface ($\xi = 0$), the left and bottom boundaries of the domain ($\xi = 1$) and the right and top boundaries ($\xi = 0$). The system is allowed to reach steady-state and the L_2-norm of error in temperature over the whole domain is calculated. Both FD + LS and FV + MT methods are used to solve this problem. Variation of the L_2 norm with grid spacing using the two methods is compared in Fig. 5(b) by taking the solution on the finest mesh (200×200) to be the exact solution. The figure shows that the present FD + LS solution compares well with the previous FV + MT method in terms of the order of accuracy. Both methods provide globally second-order accurate solutions although the errors in the finite difference case are somewhat higher in magnitude than the FV + MT case.

The scalar convection–diffusion equation is solved next in the same setup as for the above case but a uniform flow ($u = 1.0$, $v = 1.0$) is imposed everywhere in the domain. As in the previous case, both FD + LS and FV + MT methods are used to solve the convection–diffusion equation. The variation of L_2 error norms with grid spacing for the solution at steady-state for each of these methods is plotted in Fig. 5(c). The slopes of the lines indicate that both methods yield second-order accurate results. These accuracy studies indicate that the present FD + LS method compares well with the finite-volume sharp interface method while carrying the advantage of simplicity of construction.

8.2. Choosing a solver: evaluating the performance of AMG

The performance of the algebraic multigrid solver is evaluated for typical problems with embedded sharp interfaces which involve solid–fluid or fluid–fluid interfaces or both. The treatment of the solid–fluid

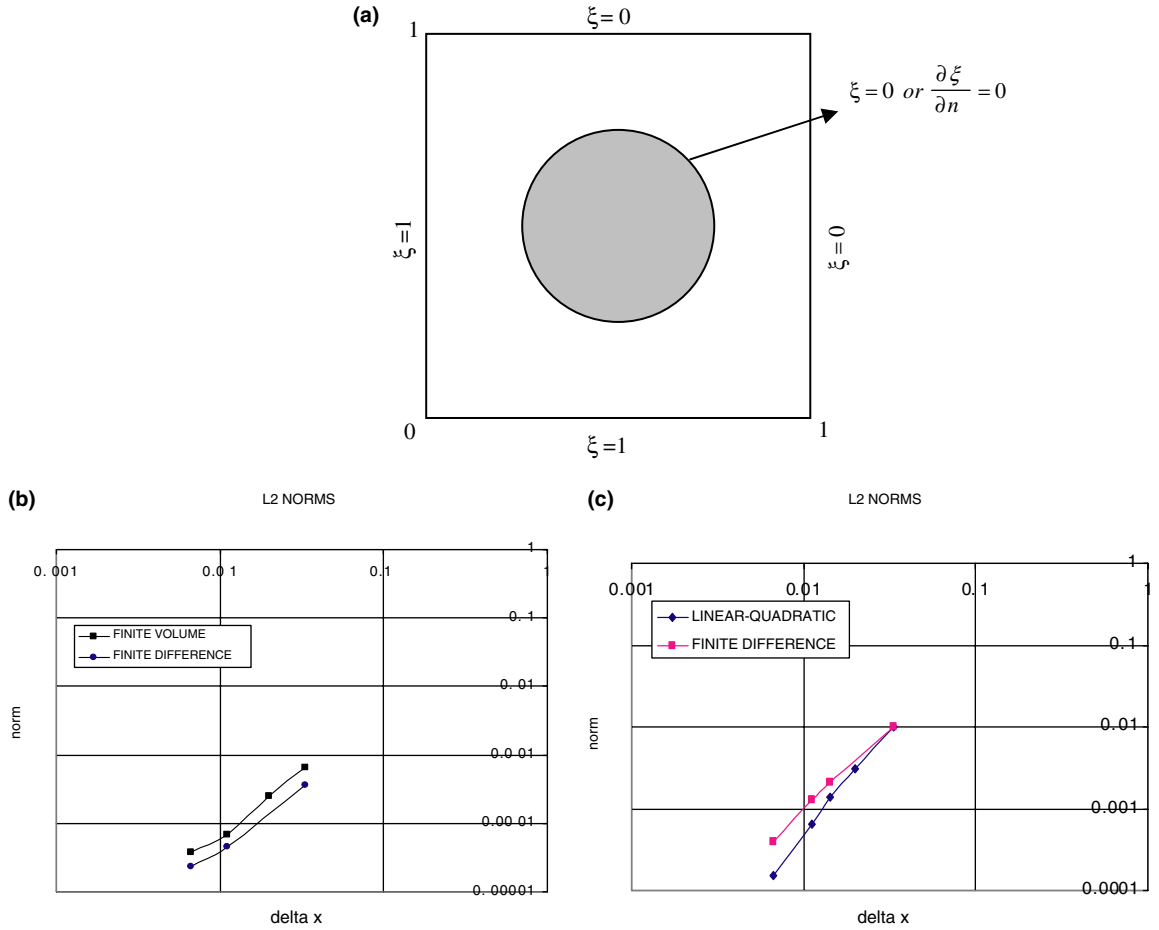

Fig. 5. Comparison of error norms for finite-difference and finite-volume formulations: (a) computational setup; (b) diffusion problem; (c) convection–diffusion problem.

interfaces has been detailed in this paper and the fluid–fluid interface is handled using the Ghost Fluid Method [31]. Further details on combining the fluid–fluid interface with solid–fluid interfaces, including treatment of contact and wetting, are provided in a companion paper [2]. An important consideration while selecting a solver is the scalability of the method. The scalability of a method is defined as the ratio of the solve time to the number of computational points on mesh. With increasing number of computational points, the performance of a method with better scalability gets increasingly better when compared to other solvers [68,69]. The number of work units in the solution process is defined by

$$\varpi = \sum_{m=1}^{M} n_m \frac{N_m}{N_1}, \tag{49}$$

where ϖ indicates the number of work units and n_m denotes the number of inner iterations on level m of mesh, N_m is the number of points on level m of the mesh and N_1 is the number of points on the finest level of mesh. This is a measure of the computational effort expended in the solution of a given problem. The CPU time required for the solution to converge to a residual level of 1e − 6 in the L_∞ norm is measured

for the advancement of the solution for 25 time steps starting from an arbitrary initial condition for the pressure and velocity fields. For the stationary boundary problems considered below, at the end of 25 time steps, the effects of the arbitrary initial conditions have generally been overcome and the subsequent convergence is fairly rapid. Because of the presence of embedded boundaries and the non-uniformity of coefficients of the cells adjacent to the interface, the highest values of residuals are generally concentrated close to the interfacial cells. The data presented have been recorded for a maximum of 3 levels of multigrid so as to ensure uniformity. More levels can be used as well depending on the number of mesh points on the finest mesh and the aspect ratio of the flow domain. The performance of the AMG solver is tested below for a hierarchy of problems involving embedded interfaces. The performance is compared with the standard point wise Gauss–Siedel and Line-SOR solvers.

8.2.1. Scalability tests

The scalability characteristics of the three solvers tested are shown in Table 1 for different applications with and without embedded interfaces. In each case, the scalability characteristic of the AMG solver is found to be at least three times better than that of the other solvers depending on the specific application. The relative scalability of the solvers is more or less independent of the type of application (with or without interfaces) as seen from Table 1. With increasing grid size the performance of AMG gets increasingly better when compared to the line solver and Gauss–Siedel point solver. For instance, the data for the flow across cylinder in 2D in Table 1 shows a scalability of 0.052 for the line solver, 0.054 for the point solver and 0.01 for the multigrid.

8.2.2. Solve time comparison

Table 2 shows the comparison of solver parameters (the solve time, work units and number of fine mesh iterations) on a variety of moving boundary problems with immersed interfaces. The problem specifications are listed in the table. The recorded data is scaled by the corresponding values for a Gauss–Siedel point solver. The AMG solver is found to consistently yield a speed-up of 5 times over the point solver in each case, irrespective of the type of the immersed interface. The Line solver on the other hand performs better in the case of solid–fluid interfaces than fluid–fluid interfaces.

8.2.3. Moving boundaries with local coarsening solid–fluid and fluid–fluid interface

This case involves moving solid–fluid as well as fluid–fluid interfaces and tests the performance of the AMG solver with multiple embedded boundaries with and without local coarsening. A bubble is placed in a tube with moving walls. The tube contracts and expands periodically. The bubble is deformed and transported due to the contraction of the tube. The evolution of the shape of the tube walls and the bubble

Table 1
Comparison of the scalability of different solvers with increasing grid-size

Problem	Scalability (s/computational point)		
	Gauss–Siedel point solver	Line solver	Algebraic multigrid
Channel flow in 2D ($Re = 250$): no interfaces	0.045	0.05	0.009
Flow across cylinder in 2D ($Re = 50$): solid–fluid interface	0.052	0.054	0.01
Flow across sphere in 3D ($Re = 100$): solid–fluid interface	0.07	0.06	0.02
Bubble rising in gravity ($Re = 526$, $We = 1.5$, $\rho_w/\rho_a = 1000$): fluid–fluid interface	0.2	0.4	0.07

The solve time is plotted against the number of computational points in the domain. This is a straight line, the slope of which indicates the scalability of the method (Webster [68,69]). A method with better scalability has a lower slope. This indicates that with increasing number of computational points, the performance of a method with better scalability gets better when compared to other solvers. As seen in the above table, the scalability of the AMG solver is better than that of the other solvers for problems with and without embedded interfaces.

Table 2
Quantitative comparison of solver data for different applications

Problem	Solver					
	Line solver			Algebraic multigrid		
	Solve time	Work units	Fine mesh iterations	Solve time	Work units	Fime mesh iterations
Flow across cylinder in 2D ($Re = 20$)	0.2720	0.1871	0.1871	0.2008	0.0744	0.0434
Flow across cylinder in 2D ($Re = 300$)	0.3265	0.1868	0.1868	0.1916	0.0717	0.0415
Bubble rising in gravity ($Re = 526$, $We = 1.5$, $\rho_w/\rho_a = 1000$)	0.8709	0.6566	0.6566	0.1195	0.0476	0.0290
Bubble evolution under the action of a peristaltic wave ($Re = 10$, $We = 0.01$, $\rho_w/\rho_a = 100$)	0.8224	0.7048	0.7048	0.3092	0.2067	0.1299
Flow across sphere in 3D ($Re = 100$)	0.5170	0.1762	0.1762	0.3269	0.1015	0.0667

The recorded data is scaled by corresponding values for a Gauss–Siedel point solver. For example, the reported value for solve time is the ratio of solve time for Line solver or the algebraic multigrid solver to the solve time for Gauss–Siedel point solver. The other values are reported in a similar way.

in the contraction phase are shown in Fig. 6(a). As seen in Table 2, AMG achieves a speed up of about 10 times that of the point solver for this problem, even in the presence of multiple moving interfaces. Fig. 6(b) illustrates the recorded solver data for this application. The figure shows the comparison of solve time, work units, fine mesh iterations tabulated in Table 2 as well as the relative set-up times for the multigrid with and without local coarsening. While there is a 10 times speed-up in the set-up process with local coarsening the solve times in both cases are comparable. This shows that in the case of moving boundary problems, where the interface coefficients change at every time step, local coarsening is attractive to save on time required for multigrid set-up. For long computational times, the saving in the set-up time leads to a large saving in total computation time when local coarsening strategy is used.

8.3. Benchmarking the flow solver

8.3.1. Two-dimensional flows

(a) *Flow past a stationary circular cylinder:* This study validates the current FD + LS method for flow around immersed stationary boundaries over a range of Reynolds numbers by simulating steady and unsteady flows past a circular cylinder immersed in an unbounded flow. These 2D cases are used to directly compare the current FD + LS technique with experimental data as well as benchmarked numerical results, including the previous FV + MT technique [29]. The flow around a cylinder for Reynolds number (Re) = 40, 80, and 300 are simulated and the results are compared to various numerical and experimental results. A large domain size of 30×30 with computational mesh of 452×452 has been used to minimize the boundary effects and to resolve the boundary layer on the embedded solid. The grid is fine and uniform in the vicinity of the immersed boundary and is stretched linearly away from it. The boundary conditions on the domain boundaries correspond to potential flow past a cylinder.

The length of the recirculation, the drag coefficient, $C_D = F_D/(1/2)\rho U_0^2 D$ (F_D being the drag force) and the Strouhal number are computed for comparison with benchmark results. Table 3 shows that the results from the current FD + LS method compare well with other experimental and numerical solutions. For $Re = 40$, the flow develops to a steady-state with a steady recirculation zone of length 1.48 behind the cylinder. The unsteadiness in the flow manifests itself in the form of classical Karman vortex street at $Re = 80$. A Strouhal number ($St = fD/U_0$) of 0.15 is calculated for this flow. The mean drag coefficient computed is 1.37 which is in agreement with the previous FV + MT results. A similar study is conducted by increasing the Reynolds number up to 300. The values of mean drag coefficient $C_D = 1.28$ and $St = 0.21$ agree well with the experimental value of [70] as well as previous FV + MT results. Thus, for flows with embedded

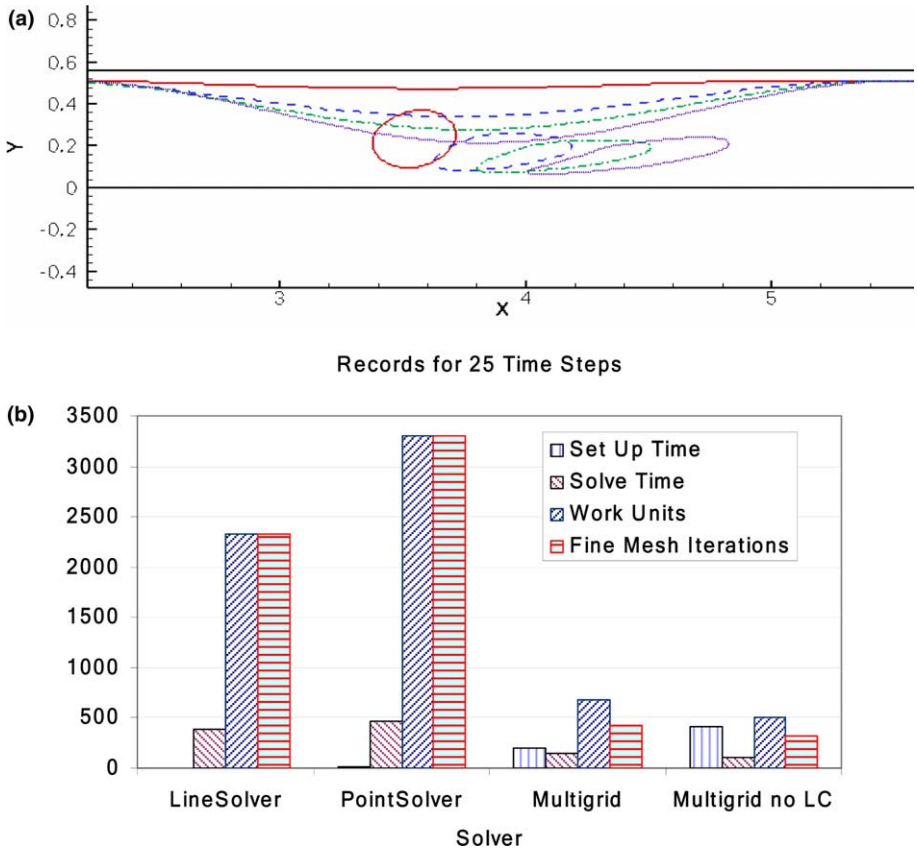

Fig. 6. Solid–fluid–fluid moving interface problem. (a) Bubble evolution when subjected to peristaltic wave in a channel. $Re = 1$, $We = 0.01$, $\rho_w/\rho_a = 100$. (b) Comparison of solver data for 25 time steps. LC, local coarsening.

boundaries the current FD + LS method yields results that are indistinguishable from the FV + MT calculations.

(b) *Flow past an oscillating cylinder:* To validate the capability of handling flows past moving boundaries the flow past a transversely oscillating cylinder and the classical "lock-on" phenomenon for vortex shedding is examined. Flow past a cylinder at $Re = 200$ undergoing sinusoidal transverse oscillation is simulated for a range of oscillation frequencies and the results are compared to the experimental results [71].

The cylinder center (x_0,y_0) is located at (8,10) relative to the left bottom corner in a 20×20 domain with 400×400 mesh points. A uniform velocity U_0 is prescribed on the left, top and bottom boundaries and the right boundary has an outlet boundary condition. A sinusoidal motion of the form $x_c(t) = x_0$, $y_c(t) = y_0 + A\sin(2\pi f_f t)$ is imposed on the cylinder, where t is the non-dimensional time, A is the amplitude of oscillation, f_f is the frequency of the imposed oscillation.

The unsteady flow past a stationary cylinder at $Re = 200$ is used as a starting solution before proceeding to oscillate the cylinder. A vortex-shedding frequency of $f_0 = 0.198$ and drag coefficient of $C_D = 1.37$ are calculated for this flow. With the solution from the above simulation as an initial condition, flow past an oscillating cylinder for a range of oscillation amplitudes and frequencies has been simulated. The sets of parameters chosen to study the lock-on phenomenon are: (a) $A = 0.1$, $f_f = \{0.15, 0.17, 0.19, 0.2, 0.21, 0.23, 0.25\}$ and (b) $A = 0.2$, $f_f = 0.15, 0.17, 0.19, 0.21$. Each of the simulations was performed for about 200 non-dimensional time units to ensure that the flow has attained a quasi-steady state. Shown in Fig.

Table 3
Comparison with benchmark data for flow around cylinder

Study	*Re*					
	40		80		300	
	C_D	L/D	C_D	St	C_D	St
Tritton [77]	1.48	–	1.29	–	–	–
Fornberg [78]	1.50	2.24	–	–	–	–
Mittal and Balachandar [79]	–	–	–	–	1.37	0.21
Williamson [80]	–	–	–	0.15	–	0.20
Finite volume [43]	1.52	2.27	1.37	0.15	1.38	0.21
Current	1.52	2.30	1.36	0.15	1.28	0.22

7(a) are the vortex shedding patterns behind a cylinder oscillating with amplitude of 0.1 and frequency of 0.21. The non-dimensional shedding frequency (Strouhal number) is calculated by evaluating the periodicity of the velocity fluctuations in the near wake region. The variation of velocity at a probe point in the wake for the above mentioned case is shown in Fig. 7(b). The deduced shedding frequency from this plot is 0.21, which implies that the vortices shed by the cylinder are "locked-on" to the cylinder oscillation. Based on experiments [71], Koopmann obtained a curve in amplitude vs. frequency parameter space that demarcates the lock-on region from non-lock-on regions. The results from the present study are benchmarked by verifying the lock-on prediction against Koopmann's experimental curve. The current results are compared with the experimental curve in Fig. 7(c) and show good agreement. The current simulations predict lock-on at frequencies close to the natural frequency. The points where lock-on was observed (filled-squares) fall within the experimental lock-on regime (region between the solid lines in Fig. 7(c)) thereby validating the current technique.

8.3.2. Three-dimensional flows

(a) *Flow around a stationary sphere:* Simulation of flow around a stationary sphere is used to validate the numerical method for problems involving three-dimensional fixed immersed boundaries. Benchmark numerical solutions and experimental data are used to validate the solutions. There are distinct regimes in the flow around a sphere that can be identified based on the Reynolds number of the imposed flow [72]. These include the transition from steady axisymmetric to non-axisymmetric flow (at $Re \cong 210$), followed by transition from steady to unsteady flow (at $Re \cong 270$). Laminar flows up to $Re = 300$ are simulated and compared to benchmark data. The transition points as well as quantitative measures such as recirculation length, drag coefficient and Strouhal number (for unsteady flows) are used to validate the current method. A $15 \times 15 \times 15$ domain with the sphere centered at ($x = 5$, $y = 7.5$, $z = 7.5$) is used for the simulations in the current study. A computational grid consisting of $130 \times 110 \times 110$ mesh points with fine and uniform grid in the vicinity of the sphere and linearly stretched mesh away from the sphere is used. Computational resources limit the domain and mesh size in 3D transient calculations. The domain and mesh size used are deemed sufficient for the present study based on the agreement between the present results and benchmarks. The boundary conditions on all the boundaries correspond to potential flow past a sphere.

Steady axisymmetric flow: The flow around a sphere is steady and separated for Reynolds numbers in the range 20–210 [72]. The flows for Re starting from 50 are simulated. Fig. 8(a)–(d) illustrates the streamlines, through the symmetry plane ($z = 7.5$) plane for Reynolds numbers of 50, 100, 150 and 200, respectively. As expected, the flow has a steady, axisymmetric wake in all these cases. For these Reynolds numbers, the quantitative data, viz. length of the recirculation bubble and the drag coefficient tabulated in Table 4 compare well with the published data in the literature.

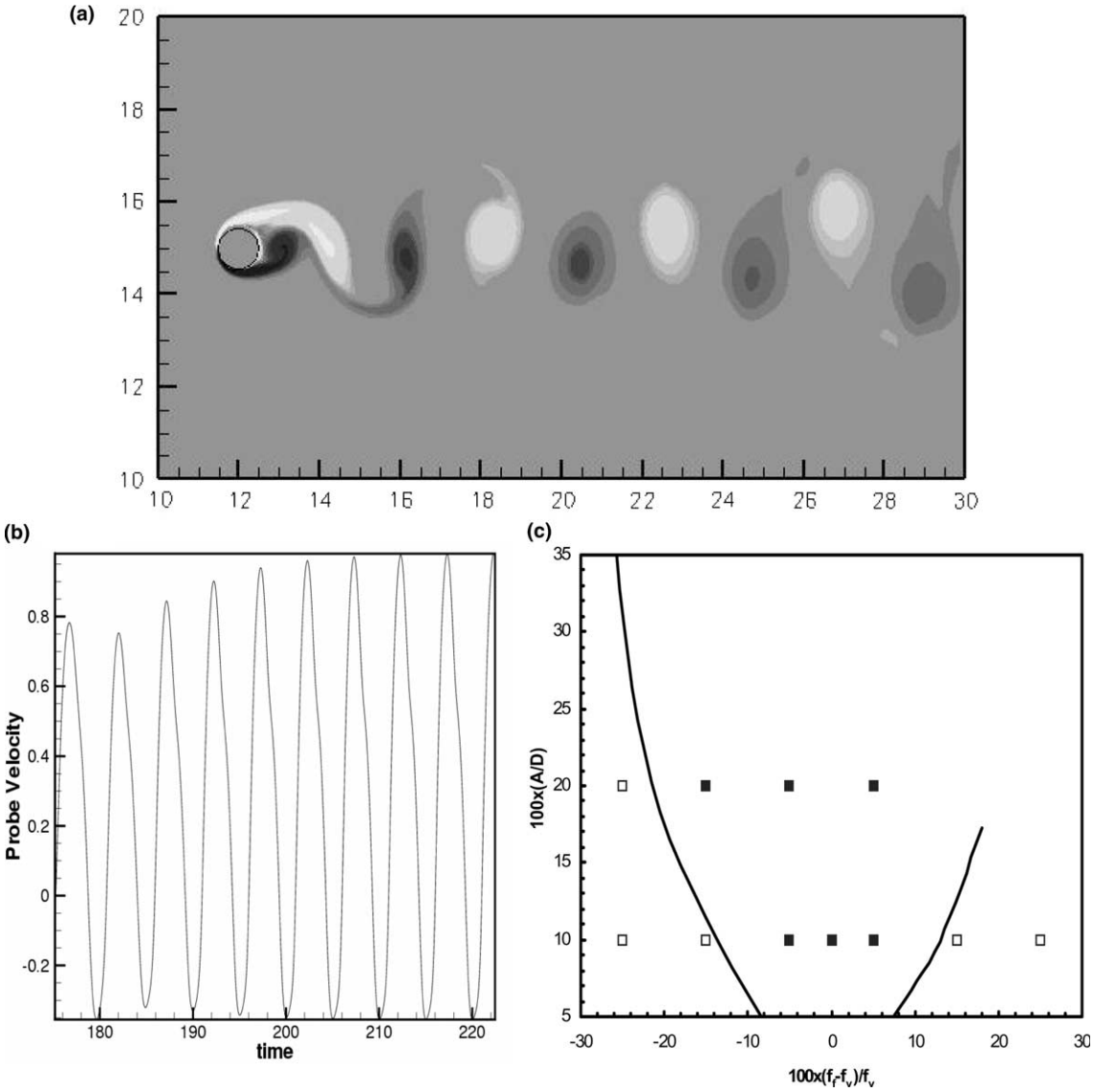

Fig. 7. Flow past an oscillating cylinder at $Re = 200$: (a) Spanwise velocity contours. (b) Fluctuations in spanwise velocity component at a point in the wake region. (c) Comparison of the present results with the Koopmann curve for lock-on region. The closed squares indicate lock-on frequencies and open square indicate no lock-on.

Steady non-axisymmetric flow: It has been experimentally established [73] that a non-axisymmetric steady flow regime prevails in the range $210 < Re < 270$. Numerical results [72] indicate the value for onset of the asymmetry to be close to $Re = 215$. The onset of this flow regime has been verified by simulating flows for $Re = 210$ and $Re = 215$. The plots in Fig. 8(d) and (e) show the streamlines and the pressure contours on $z = 7.5$ plane for a $Re = 210$ and $Re = 215$, respectively. At $Re = 215$, the flow field indicates an incipient axial asymmetry which is highlighted by the circled region in Fig. 8(e). This establishes the transition to asymmetric flow in this Re interval.

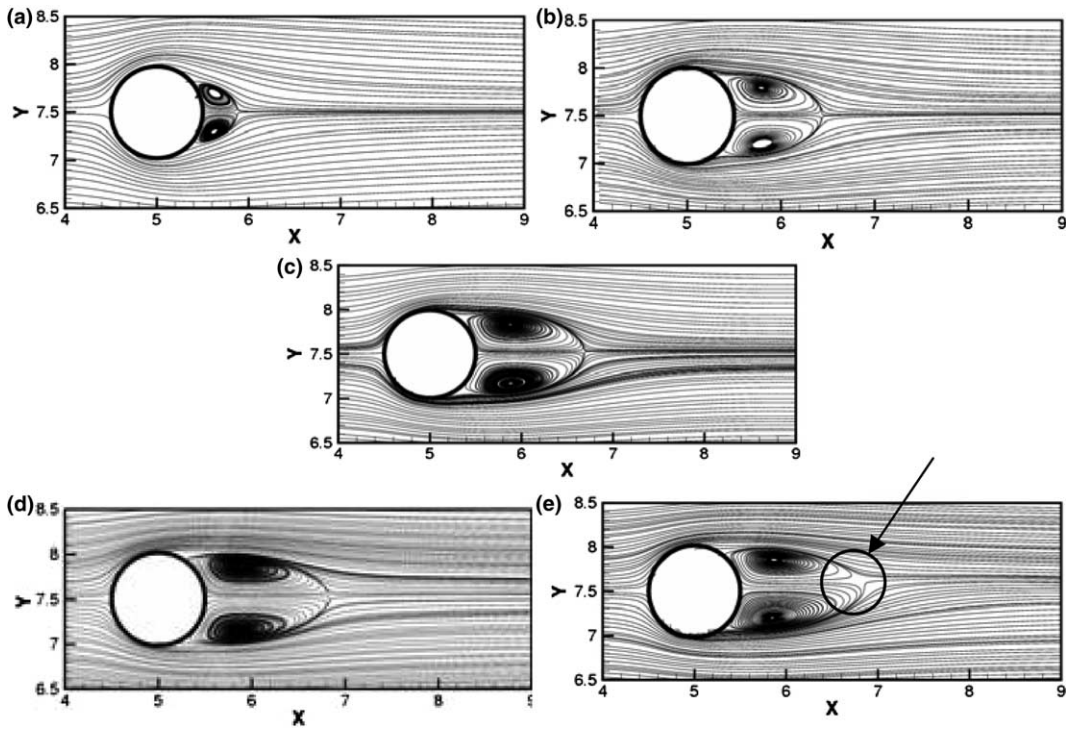

Fig. 8. The axisymmetric streamlines past the sphere: (a) $Re = 50$; (b) $Re = 100$; (c) $Re = 150$; (d) $Re = 225$, u,v vectors on the x–y plane; (e) u,w vectors on x–z plane.

Unsteady flow: The plots in Fig. 9(a)–(c) show the flow features on the $z = 7.5$ plane at $Re = 270$. As can be observed from the vorticity contours in this graph, the flow is still steady in this case. Since the transition to unsteady flow is known to occur in $270 < Re < 300$ range [73,74], flows are computed within this range by varying Re in intervals of 10. A steady solution from $Re = 270$ is employed as an initial condition for all the test simulations. The vorticity contours on various planes shown, for $Re = 280$, in Fig. 9(d)–(f) reflect the unsteadiness in the flow at that Reynolds number.

Calculations for $Re = 300$ were carried out to observe the well-established vortex shedding phenomenon and also to obtain quantitative data for comparison with benchmarks. Several techniques can be used for identifying and extracting vortical structures in 3D. The Δ-method proposed by Chong et al. [75] has been used for visualizing the vortical structures in the present investigation. The above method defines the vortex

Table 4
Comparison of results with benchmark data for flow around a sphere

Study	*Re*									
	50		100		150		215		300	
	C_D	L/D	C_D	L/D	C_D	L/D	C_D	L/D	C_D	St
Mittal [81]	1.57	0.44	1.09	0.87	–	–				
Clift et al. [82]	1.57	–	1.09	–	0.89	–	0.74			
Johnson and Patel [72]	1.57	0.40	1.08	0.86	0.90	1.20			0.629	0.137
Current	1.56	0.39	1.06	0.88	0.85	1.19	0.70	1.31	0.621	0.133

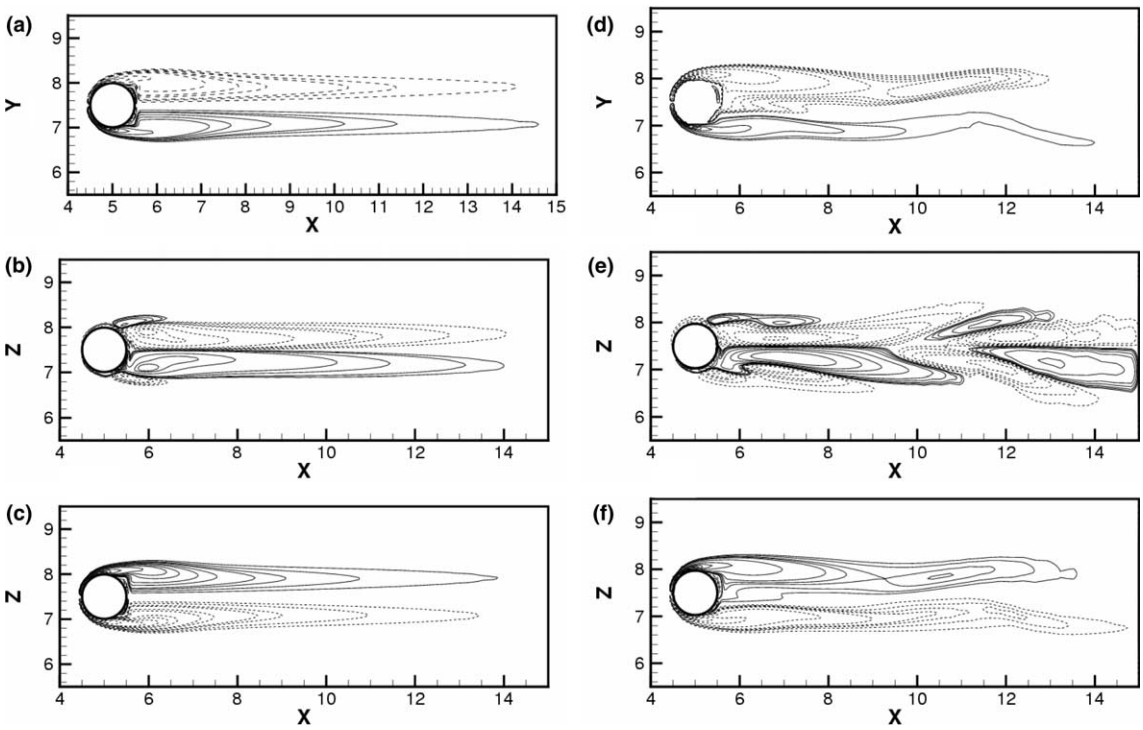

Fig. 9. Vorticity contours for $Re = 270$ [(a) ω_z on x–y plane; (b) ω_x on x–z plane; (c) ω_y on x–z plane] and $Re = 280$ [(d) ω_z on x–y plane; (e) ω_x on x–z plane; (f) ω_y on x–z plane].

core as a region with closed or spiral local streamline pattern. Such a region is identified by complex eigenvalues of velocity gradient tensor ∇u. The iso-surfaces of complex eigenvalues of ∇u are plotted in Fig. 10. The structure of the vortices shed from the sphere resemble the numerical results in [72]. The calculated average drag coefficient of 0.621 as well as the Strouhal number of 0.133 compare well with results of Johnson and Patel [72] (Table 4).

(b) *Flow around a moving sphere:* Flow around a sphere undergoing stream-wise oscillations in a stably stratified fluid has been studied to validate the current numerical method for three-dimensional flows involving moving embedded objects. The computational setup is the same as in the stationary sphere cases presented in the previous section. The sphere undergoes forced oscillation in the stream-wise direction, $U(t) = U_1 \cos f_f t$, where U_1 and f_f are the amplitude and frequency of oscillation. The governing equations for the flow are the same as discussed in Section 3.2. The fluid is stably stratified with an undisturbed linear density gradient in the z-direction, $\rho_B(z) = \rho_{H/2} + (\Delta\rho/2)(1 - 2z/H)$, where $\rho_{H/2}$ is the density at the mid-depth and $\Delta\rho$ is the density difference between the bottom ($z = 0$) and top ($z = H$) planes of the domain. The terms involving density variations assume significance due to stratification. The density field is computed using incompressibility condition, Eq. (5). The presence of the oscillating sphere introduces additional non-dimensional quantities, viz., the Keulegan–Carpenter number ($KC = U_1/f_f D$), which characterizes the non-dimensional amplitude of the oscillations, and normalized forcing frequency ($S_f = f_f D/(2\pi U_0)$).

Stratified flows display a range of interesting phenomena usually not found in the homogeneous flow situations. For flow around a sphere, Lin et al. [52] have shown experimentally that for sufficiently small Fr or strong stratification, the incoming flow prefers to go around the sphere rather than over it. For $Fr < 0.2$, the flow is quasi-two-dimensional. The stratification suppresses the formation of hairpin vortices at higher

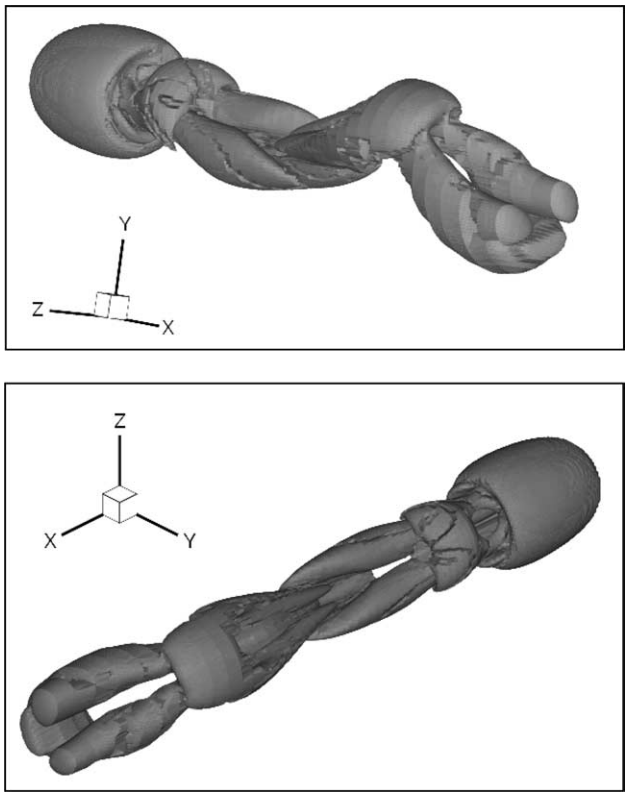

Fig. 10. Vortical structures for flow at $Re = 300$. Oblique views.

Reynolds numbers. For sufficiently large Re, vortex shedding in the lee of the sphere resembles the classic Karman vortex street observed for uniform flow past a circular cylinder and the flow becomes nearly two-dimensional in $-0.5 < z/D < 0.5$. The wide range of experimental results reported by Lin et al. [52,76] for these parameter values serve for validation. The parameter values chosen for the current study are $Re = 190$ and $Fr = 0.07$. Lin et al. [52,76] created a flow regime diagram in KC $-$ S_f parameter space. The test points chosen for the current study are labeled A (KC = 0.03, $S_f = 0.35$) and B (KC = 0.2, $S_f = 0.35$). Experimental results indicate that test points A and B fall into the natural vortex shedding and lock-on alternate single vortex regimes, respectively. The simulations of flow around an oscillating sphere are performed to test of the flow behavior predicted agrees with the experimental flow regime diagram.

The flow around a stationary sphere at $Re = 190$ and $Fr = 0.07$ is first simulated and the solution is used as an initial condition for the moving sphere cases. The vortex structures in the lee of the sphere resemble the classical Karman vortex street observed in the case of a circular cylinder, as seen from the vorticity contours on the central $z = 7.5$ plane shown in Fig. 11(a). The contours of the z-component of velocity and the density on the $z = 7.5$ plane are plotted in Fig. 11(b) and (c). The small variations in the plane in these plots illustrate the essentially quasi-two-dimensional nature of this flow. A Strouhal number of 0.2 calculated from the periodicity of the velocity for a probe point in the lee of the sphere matches with experimental value reported in the literature [52].

The quasi-steady state solution of the above simulation is used as the starting condition for the oscillating sphere cases. For the first test point A, the amplitude of oscillation (KC = 0.03) is small, so that the far wake shedding behavior is similar to that of the stationary case. The Strouhal number, calculated to be 0.2,

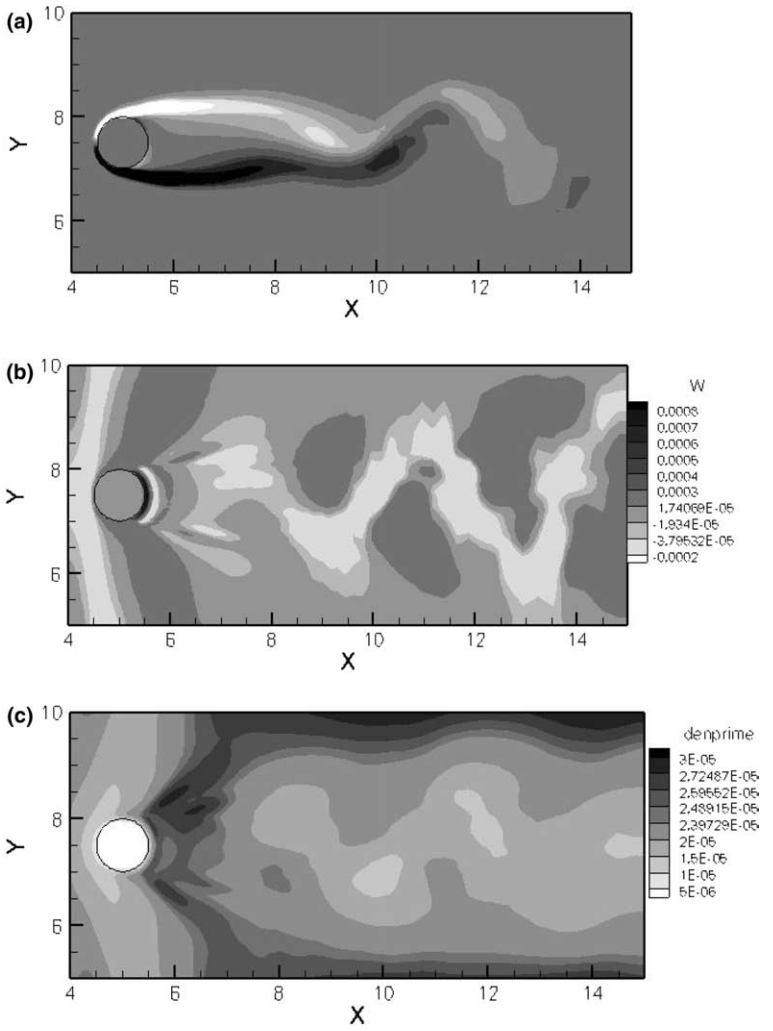

Fig. 11. Flow characteristics for $Re = 190$ and $Fr = 0.07$, KC = 0.0, $S_f = 0.0$ [(a) vorticity, ω_z on x–y plane; (b) w velocity contours on x–y plane; (c) density contours on x–y plane].

is unchanged from the natural shedding frequency of the stationary sphere. The flow being similar to the stationary case, this flow regime is classified as being in the ''natural vortex shedding'' regime.

For test point B, at a higher KC of 0.2, the vortical structures visualized using the Δ-method (as described in Section 8.3.2) are plotted in Fig. 12. The cylindrical vortices apparent in the x–y view shown in Fig. 12(b) illustrate that the vortex shedding is self similar in z-direction. This demonstrates the quasi-two-dimensionality imposed due to stratification. The x–z view of vortex shedding shown in Fig. 12(c) supports the above observation. The Strouhal number calculated from the probe velocity variation with time is 0.35 which indicates that the shedding under these conditions locks on with the oscillation frequency of the sphere. Fig. 12 also illustrates that the vortices are being shed alternately from either side of the sphere. Hence this flow regime is named ''lock-on alternate single vortex'' regime [76]. The flow structures viewed from the various perspectives are visually identical to those obtained in the experiments of Lin et al. [76]. In summary, flow around an oscillating sphere in the presence of density stratification has been simulated for

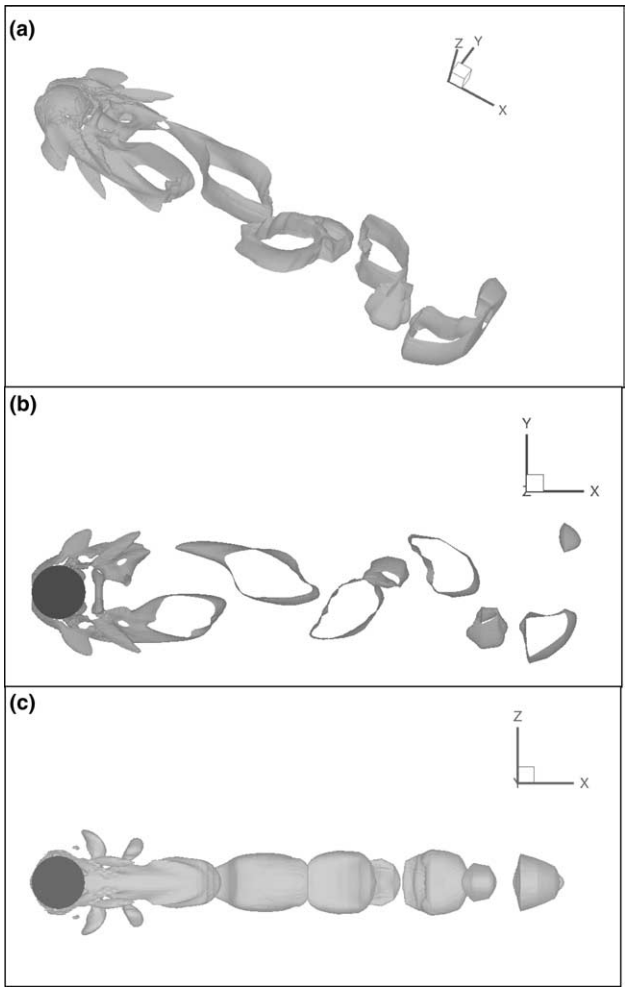

Fig. 12. Vortical structures for the flow around oscillating sphere flow at $Re = 190$, $Fr = 0.07$, KC = 0.2, $S_f = 0.35$: (a) oblique view; (b) x–y view; (c) x–z view.

selected set of parameters KC and S_f. The flow regimes, vortical patterns and shedding frequencies predicted by the current technique for these test points are in agreement with the experimental results reported by Lin and coworkers and thereby provide validation for the present methodology for three-dimensional flows involving moving immersed objects.

9. Conclusions

A numerical technique has been developed to solve incompressible fluid flows with immersed bodies in two and three-dimensions. The salient features of the technique are a globally second-order accurate finite-difference discretization, a special treatment for the points lying adjacent to the immersed interface to capture the interface features in a sharp fashion and a level-set interface representation. The method does not employ forcing functions in the governing equations. Instead the discretization is performed in such a way

that the interface boundary conditions are incorporated into the discrete system of equations at the interface-adjacent grid points. In contrast to a previous finite-volume sharp interface method the discretization procedure is quite simple and entails only a modest modification of a simple Cartesian grid flow solver. The simplicity is facilitated by the level-set description of the interface. An algebraic multigrid method has been adapted to include moving immersed solid–fluid and fluid–fluid interfaces in order to accelerate the solution of the discrete pressure Poisson equation.

Several computational tests have been carried out to benchmark the method. The results from the accuracy studies, in which the present method is compared to a finite-volume method, show that the order of accuracy is not compromised by the present finite-difference method. Validation for two-dimensional flows around stationary and oscillating cylinders has been performed. The current results match well with the benchmark results. Flow around a stationary sphere has been simulated for a range of Reynolds numbers to validate the three-dimensional capability. Flow around an oscillating sphere has been computed as a demonstration of the ability to handle three-dimensional moving boundaries. The results in the three-dimensional cases compare well with the established experimental and numerical results.

Acknowledgments

This work was performed with support from the Computational Mechanics Branch, AFRL-MNAC, Eglin, FL (Project manager Joel Stewart) and AFOSR Computational Mathematics Division (Program manager Fariba Fahroo).

References

[1] D.M. Anderson, G.B. McFadden, A.A. Wheeler, Diffuse-interface methods in fluid mechanics, Annu. Rev. Fluid Mech. 30 (1998) 139–165.
[2] H. Liu, et al., Sharp interface Cartesian grid method II: a technique for simulating droplet interactions with surfaces of arbitrary shape, J. Comput. Phys., (accepted).
[3] Y. Yang, H.S. Udaykumar, Sharp interface Cartesian grid method III: solidification of pure materials and binary solutions, J. Comput. Phys., (accepted).
[4] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys 25 (220–243) (1977).
[5] G. Tryggvason, et al., Computations of multiphase flows, Adv. Appl. Mech. 39 (2003) 81–120.
[6] S.V. Marella, H.S. Udaykumar, Computational analysis of the deformability of leukocytes modeled with viscous and elastic structural components, Phys. Fluids 16 (2) (2004) 244–264.
[7] H.S. Udaykumar, R. Mittal, W. Shyy, Computation of solid–liquid phase fronts in the sharp interface limit on fixed grids, J. Comput. Phys. 153 (2) (1999) 535–574.
[8] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible 2-phase flow, J. Comput. Phys. 114 (1) (1994) 146–159.
[9] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface-tension, J. Comput. Phys. 100 (2) (1992) 335–354.
[10] N. Al-Rawahi, G. Tryggvason, Numerical simulation of dendritic solidification with convection: two-dimensional geometry, J. Comput. Phys. 180 (2) (2002) 471–496.
[11] R.J. Leveque, Z.L. Li, The immersed interface method for elliptic-equations with discontinuous coefficients and singular sources, Siam J. Numer. Anal. 31 (4) (1994) 1019–1044.
[12] T. Cheng, C.S. Peskin, Stability and instability in the computation of flows with moving immersed boundaries – a comparison of 3 methods, Siam J. Sci. Stat. Comput. 13 (6) (1992) 1361–1376.
[13] J.M. Stockie, B.R. Wetton, Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes, J. Comput. Phys. 154 (1) (1999) 41–64.
[14] M.C. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2) (2000) 705–719.
[15] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (2) (1999) 509–534.

[16] L. Lee, R.J. Leveque, An immersed interface method for incompressible Navier–Stokes equations, Siam J. Sci. Comput. 25 (3) (2003) 832–856.
[17] R. Glowinski, et al., A distributed Lagrange multiplier/fictitious domain method for flows around moving rigid bodies: application to particulate flow, Int. J. Numer. Meth. Fluid 30 (8) (1999) 1043–1066.
[18] D. Adalsteinsson, J.A. Sethian, A fast level set method for propagating interfaces, J. Comput. Phys. 118 (2) (1995) 269–277.
[19] N.A. Patankar, et al., A new formulation of the distributed Lagrange multiplier/fictitious domain method for particulate flows, Int. J. Multiphase Flow 26 (9) (2000) 1509–1524.
[20] X.D. Wang, W.K. Liu, Extended immersed boundary method using FEM and RKPM, Comp. Meth. Appl. Mech. Eng. 193 (12–14) (2004) 1305–1321.
[21] E.A. Fadlun, et al., Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (1) (2000) 35–60.
[22] E. Balaras, Modeling complex boundaries using an external force field on fixed Cartesian grids in large-eddy simulations, Comp. Fluid 33 (3) (2004) 375–404.
[23] Y.H. Tseng, J.H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2) (2003) 593–623.
[24] J. Kim, D. Kim, H. Choi, An immersed-boundary finite-volume method for simulations of flow in complex geometries, J. Comput. Phys. 171 (1) (2001) 132–150.
[25] M.C. Lai, Z.L. Li, A remark on jump conditions for the three-dimensional Navier–Stokes equations involving an immersed moving membrane, Appl. Math. Lett. 14 (2) (2001) 149–154.
[26] Z.L. Li, M.C. Lai, The immersed interface method for the Navier–Stokes equations with singular forces, J. Comput. Phys. 171 (2) (2001) 822–842.
[27] R.J. Leveque, Z.L. Li, The immersed interface method for elliptic-equations with discontinuous coefficients and singular sources, Siam J. Numer. Anal. 32 (5) (1995) 1704–1704 (31 (4) (1994) 1019–1044).
[28] H.S. Udaykumar, S. Marella, S. Krishnan, Sharp-interface simulation of dendritic growth with convection: benchmarks, Int. J. Heat Mass Transf. 46 (14) (2003) 2615–2627.
[29] H.S. Udaykumar, R. Mittal, P. Rampunggoon, Interface tracking finite volume method for complex solid–fluid interactions on fixed meshes, Commun. Numer. Meth. Eng. 18 (2) (2002) 89–97.
[30] H.S. Udaykumar, et al., A sharp interface Cartesian grid method for simulating flows with complex moving boundaries, J. Comput. Phys. 174 (1) (2001) 345–380.
[31] R.P. Fedkiw, et al., A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (2) (1999) 457–492.
[32] X.D. Liu, R.P. Fedkiw, M.J. Kang, A boundary condition capturing method for Poisson's equation on irregular domains, J. Comput. Phys. 160 (1) (2000) 151–178.
[33] J. Chessa, T. Belytschko, An extended finite element method for two-phase fluids, J. Appl. Mech. Trans. Asme 70 (1) (2003) 10–17.
[34] J. Chessa, P. Smolinski, T. Belytschko, The extended finite element method (XFEM) for solidification problems, Int. J. Numer. Meth. Eng. 53 (8) (2002) 1959–1977.
[35] N. Sukumar, et al., Modeling holes and inclusions by level sets in the extended finite-element method, Comp. Meth. Appl. Mech. Eng. 190 (46–47) (2001) 6183–6200.
[36] J. Dolbow, N. Moes, T. Belytschko, An extended finite element method for modeling crack growth with frictional contact, Comp. Meth. Appl. Mech. Eng. 190 (51–52) (2001) 6825–6846.
[37] S. Chen, et al., A simple level set method for solving Stefan problems, J. Comput. Phys. 135 (1) (1997) 8–29.
[38] F. Gibou, et al., A second-order-accurate symmetric discretization of the Poisson equation on irregular domains, J. Comput. Phys. 176 (1) (2002) 205–227.
[39] F. Gibou, et al., A level set approach for the numerical simulation of dendritic growth, J. Sci. Comput. 19 (1–3) (2003) 183–199.
[40] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, J. Comput. Phys. 202 (2) (2005) 577–601.
[41] K. Yokoi, Numerical method for a moving solid object in flows, Phys. Rev. E 67 (4) (2003).
[42] K. Yokoi, et al., Three-dimensional numerical simulation of flows with complex geometries in a regular Cartesian grid and its application to blood flow in cerebral artery with multiple aneurysms, J. Comput. Phys. 202 (1) (2005) 1–19.
[43] T. Ye, et al., An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries, J. Comput. Phys. 156 (2) (1999) 209–240.
[44] H.S. Udaykumar, L. Mao, Sharp-interface simulation of dendritic solidification of solutions, Int. J. Heat Mass Transf. 45 (24) (2002) 4793–4808.
[45] H.S. Udaykumar, L. Mao, R. Mittal, A finite-volume sharp interface scheme for dendritic growth simulations: comparison with microscopic solvability theory, Numer. Heat Transf. Part B – Fundamentals 42 (5) (2002) 389–409.
[46] A.S. Almgren, et al., A Cartesian grid projection method for the incompressible Euler equations in complex geometries, Siam J. Sci. Comput. 18 (5) (1997) 1289–1309.

[47] D. Calhoun, R.J. Leveque, A Cartesian grid finite-volume method for the advection–diffusion equation in irregular geometries, J. Comput. Phys. 157 (1) (2000) 143–180.
[48] N. Sukumar, et al., Extended finite element method for three-dimensional crack modelling, Int. J. Numer. Meth. Eng. 48 (11) (2000) 1549–1570.
[49] V. Jayaraman, H.S. Udaykumar, W.S. Shyy, Adaptive unstructured grid for three-dimensional interface representation, Numer. Heat Transf. Part B – Fundamentals 32 (3) (1997) 247–265.
[50] J.A. Sethian, Fast marching methods, Siam Rev. 41 (2) (1999) 199–235.
[51] J. Strain, Fast tree-based redistancing for level set computations, J. Comput. Phys. 152 (2) (1999) 664–686.
[52] Q. Lin, et al., Stratified flow past a sphere, J. Fluid Mech. 240 (1992) 315–354.
[53] Y. Zang, R.L. Street, J.R. Koseff, A non-staggered grid, fractional step method for time-dependent incompressible Navier–Stokes equations in curvilinear coordinates, J. Comput. Phys. 114 (1) (1994) 18–33.
[54] J.A. Sethian, Evolution, implementation, application of level set and fast marching methods for advancing fronts, J. Comput. Phys. 169 (2) (2001) 503–555.
[55] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed – algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.
[56] J.A. Sethian, P. Smereka, Level set methods for fluid interfaces, Annu. Rev. Fluid Mech. 35 (2003) 341–372.
[57] D. Adalsteinsson, J.A. Sethian, The fast construction of extension velocities in level set methods, J. Comput. Phys. 148 (1) (1999) 2–22.
[58] M. Sussman, E. Fatemi, An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, Siam J. Sci. Comput. 20 (4) (1999) 1165–1191.
[59] M. Sussman, et al., An improved level set method for incompressible two-phase flows, Comp. Fluid 27 (5–6) (1998) 663–680.
[60] D.L. Chopp, Some improvements of the fast marching method, Siam J. Sci. Comput. 23 (1) (2001) 230–244.
[61] J. Strain, A fast semi-Lagrangian contouring method for moving interfaces, J. Comput. Phys. 170 (1) (2001) 373–394.
[62] H.S. Udaykumar, W. Shyy, Simulation of interfacial instabilities during solidification. 1. Conduction and capillarity effects, Int. J. Heat Mass Transf. 38 (11) (1995) 2057–2073.
[63] R. Glowinski, et al., A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: application to particulate flow, J. Comput. Phys. 169 (2) (2001) 363–426.
[64] D.M. Anderson, G.B. McFadden, A.A. Wheeler, A phase-field model of solidification with convection, Physica D 135 (1–2) (2000) 175–194.
[65] S. Popinet, Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries, J. Comput. Phys. 190 (2) (2003) 572–600.
[66] K.F.C. Yiu, et al., Quadtree grid generation: information handling, boundary fitting and CFD applications, Comp. Fluid 25 (8) (1996) 759–769.
[67] F. Losasso, F. Gibou, R. Fedkiw, Simulating water and smoke with an octree data structure, Acm Trans. Graphics 23 (3) (2004) 457–462.
[68] R. Webster, An algebraic multigrid solver for Navier–Stokes problems, Int. J. Numer. Meth. Fluid 18 (8) (1994) 761–780.
[69] R. Webster, An algebraic multigrid solver for Navier–Stokes problems in the discrete second-order approximation, Int. J. Numer. Meth. Fluid 22 (11) (1996) 1103–1123.
[70] C. Wieselsberger, New data on laws of fluid resistance, NACA TN 84 (1922).
[71] G.H. Koopmann, The vortex wakes of vibrating cylinders at low Reynolds numbers, J. Fluid Mech. 28 (1967) 501.
[72] T.A. Johnson, V.C. Patel, Flow past a sphere up to a Reynolds number of 300, J. Fluid Mech. 378 (1999) 19–70.
[73] R.H. Magarvey, R.L. Bishop, Transition ranges for three-dimensional wakes, Can. J. Phys. 39 (1961) 1418–1422.
[74] A.G. Tomboulides, S.A. Orszag, G.E. Karniadakis, Direct and large-eddy simulation of axisymmetric wakes, AIAA Paper, 1993.
[75] M.S. Chong, A.E. Perry, B.J. Cantwell, A general classification of three-dimensional flow fields, Phys. Fluids 2 (A) (1990) 765.
[76] Q. Lin, D.L. Boyer, H.J.S. Fernando, The vortex shedding of a streamwise-oscillating sphere translating through a linearly stratified fluid, Phys. Fluids 6 (1) (1994) 239–252.
[77] D.J. Tritton, Experiments on the flow past a circular cylinder at low Reynolds number, J. Fluid Mech. 6 (1959) 547.
[78] B. Fornberg, A numerical study of steady viscous-flow past a circular-cylinder, J. Fluid Mech. 98 (JUN) (1980) 819–855.
[79] R. Mittal, S. Balachandar, Inclusion of three-dimensional effects in simulations of two-dimensional bluff body wake flows, Proceedings, 1997 ASME Fluids Engineering Division Summer Meeting.
[80] C.H.K. Williamson, Vortex dynamics in the cylinder wake, Annu. Rev. Fluid Mech. 28 (1996) 477–539.
[81] R. Mittal, A Fourier–Chebyshev spectral collocation method for simulating flow past spheres and spheroids, Int. J. Numer. Meth. Fluid 30 (7) (1999) 921–937.
[82] R. Clift, J.R. Grace, M.E. Weber, Bubbles, Drops and Particles, Academic Press, New York, 1978.