Journal of Computational Physics 229 (2010) 6427–6449

Contents lists available at ScienceDirect

Journal of Computational Physics

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

A direct-forcing embedded-boundary method with adaptive mesh refinement for fluid–structure interaction problems

Marcos Vanella [c], Patrick Rabenold [b], Elias Balaras [a,b,*]

[a] Fischell Department of Bioengineering, University of Maryland, College Park, MD 20742, USA
[b] Applied Mathematics and Scientific Computation (AMSC) Program, University of Maryland, College Park, MD 20742, USA
[c] Department of Mechanical Engineering, University of Maryland, College Park, MD 20742, USA

ARTICLE INFO

Article history:
Received 16 November 2009
Received in revised form 25 March 2010
Accepted 10 May 2010
Available online 23 May 2010

Keywords:
Cartesian grids
Finite-difference method
Adaptive mesh refinement
Immersed-boundary method
Fluid–structure interaction
Large-eddy simulation

ABSTRACT

In the present work we developed a structured adaptive mesh refinement (S-AMR) strategy for fluid–structure interaction problems in laminar and turbulent incompressible flows. The computational grid consists of a number of nested grid blocks at different refinement levels. The coarsest grid blocks always cover the entire computational domain, and local refinement is achieved by the bisection of selected blocks in every coordinate direction. The grid topology and data-structure is managed using the Paramesh toolkit. The filtered Navier–Stokes equations for incompressible flow are advanced in time using an explicit second-order projection scheme, where all spatial derivatives are approximated using second-order central differences on a staggered grid. For transitional and turbulent flow regimes the large-eddy simulation (LES) approach is used, where special attention is paid on the discontinuities introduced by the local refinement. For all the fluid–structure interaction problems reported in this study the complete set of equations governing the dynamics of the flow and the structure are simultaneously advanced in time using a predictor–corrector strategy. An embedded-boundary method is utilized to enforce the boundary conditions on a complex moving body which is not aligned with the grid lines. Several examples of increasing complexity are given to demonstrate the robustness and accuracy of the proposed formulation.

1. Introduction

The accuracy of grid-based approaches for the solution of the Navier–Stokes equations depends, to a large extent, on the quality of the computational grid. For well understood physical problems structured or unstructured grids can be carefully designed to capture the important flow features. However, in highly unsteady three-dimensional problems with moving boundaries/interfaces, locally adaptive refinement of the computational grid is desirable. Over the past decades a significant amount of work on adaptive meshing has been done in the framework of unstructured grid solution methods. The lack of inherent structure of such grids usually allows for a straightforward implementation of a variety of adaptive mesh refinement (AMR) strategies. A widely used approach is the so called *h*-refinement, where local refinement is achieved by splitting existing cells into several smaller ones, or by locally introducing additional nodes (see [28] for a review). For problems that involve moving boundaries, however, local mesh motion (*r*-refinement) is also necessary to maintain grid quality [37]. A drawback of the above methods, especially in fluid–structure interaction problems with large boundary motions and deformations, is the difficulty to control grid quality, which has an adverse impact on accuracy and stability of the computations.

* Corresponding author at: Fischell Department of Bioengineering, University of Maryland, College Park, MD 20742, USA. Tel.: +1 301 314 9477.
E-mail addresses: mvanella@umd.edu (M. Vanella), rabenold@math.umd.edu (P. Rabenold), balaras@umd.edu (E. Balaras).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.05.003

Recently, non-boundary conforming formulations have been emerging as a viable alternative to unstructured methods especially for fluid–structure interaction problems (see [30] for a review). To compute the flow around a complex body the equations of motion are usually solved on a fixed structured grid, which is almost never aligned with the body. Depending on the specifics of the formulation, boundary conditions can be imposed by appropriately modifying the stencil in the neighborhood of the body [38], or using a forcing function which can be derived either using physical arguments [33], or directly from the discrete problem [18]. Although such methods have traditionally been applied to low Reynolds number flows, recent applications on complex turbulent and transitional flows with moving boundaries have been reported with very good results [44,15]. A drawback of these formulations, however, is the lack of flexibility to selectively distribute points in areas of high velocity gradients without increasing the grid resolution in all areas of the computational box. Formulations where the grid can be refined locally could be beneficial, provided that the overall cost does not exceed that of a single-block with equivalent resolution everywhere.

Two general categories of AMR for structured grids, which can be ideally coupled to the class of non-boundary conforming methods mentioned above, can be identified: (i) isotropic splitting of individual cells that can be managed using hierarchical tree [9], or fully unstructured data-structures [21]; (ii) grid embedding, where block-structured grids composed of nested rectangular patches are used. The latter approach maintains most of the advantages of structured grid methods and is an attractive platform for introducing AMR capabilities in eddy resolving techniques such as the large-eddy simulation (LES) and direct numerical simulation (DNS) approach. Structured adaptive mesh refinement (S-AMR) was initially introduced by Berger and Oliger [14] for the solution of one- and two-dimensional hyperbolic problems, where the sharp discontinuities in the solution were better captured with increasingly refined rectangular grid patches. Since then, the method has been extended to three-dimensions, and has been demonstrated to be a robust, cost effective approach for a range of hyperbolic problems (see for example [10,13]). Applications to incompressible flows, however, have been limited, primarily due to complications associated to the enforcement of the divergence-free constraint. Most algorithms for incompressible flows are based on the extension of the second-order projection method by Bell et al. [11] on the grid topology proposed in [14]. In this particular splitting scheme the viscous and advective terms in the momentum equation are advanced via a Crank–Nicolson scheme, and an unsplit, second-order upwind Godunov method is used to evaluate the nonlinear term at the time-centered location. In most cases time refinement (i.e. different timesteps are used in different AMR levels) is also employed and synchronization of the solution between AMR levels is required to ensure the divergence-free constraint. Recent applications to multiphase flows and to prototypical laminar flows can be found [2,27] respectively. Applications of S-AMR strategies such as the above to fluid–structure interaction problems have been limited. Roma et al. [34], for example, developed an AMR version of the immersed boundary method presented in [33] for two-dimensional, viscous incompressible flows. The overall formulation is based on an implicit projection scheme inspired by the method proposed in [11], and is applied on the composite grid structure introduced by Berger and Oliger [14]. The approach was tested on a two-dimensional model problem (the two-dimensional analog of an elastic spherical balloon, filled with the same fluid present outside), and no significant difference was found between the solutions obtained on a mesh refined locally around the immersed boundary, and on a uniform mesh with the same resolution as the finest AMR level. A similar formulation for collocated grids was later developed by Griffith et al. [20].

In the present study we propose a S-AMR strategy for the solution of the Navier–Stokes equations in laminar and turbulent incompressible flows. Based on the ideas introduced by [14] a single-block solver is employed on a hierarchy of subgrids with varying spatial resolution. Each of these sub-grid blocks has a structured Cartesian topology, and is part of a tree data-structure that covers the entire computational domain. One of the main features of the present implementation is the utilization of the Paramesh toolkit [25] to keep track of the grid hierarchy, and perform the required restriction/prolongation and guard-cell filling operations. Paramesh has been used with great success to develop AMR solvers in area of magnetohydrodynamics (MHD) [24], hydrodynamics and cosmology [19], but we are not aware of applications to the solution of the Navier–Stokes equations for incompressible flows. The building block of our implementation is a single-block solver, which is utilized in each sub-block, and is a standard fractional step, finite-difference solver on a staggered grid that has been extensively used in DNS and LES of turbulent and transitional flows (see for example [6,5]). As we will demonstrate the accuracy and optimal conservation properties of the basic solver are maintained in the AMR code. To compute the flow in complex geometries a direct-forcing, embedded-boundary method is used [41].

In the next section the problem formulation is outlined. In Section 3 a description of AMR grid topology is given and the treatment of the solution at block boundaries is discussed. A strategy to address issues related to mass conservation at interfaces and grid adaptation is also proposed. Section 4 provides an overview of the fluid–structure interaction methodology employed. In Section 5 the following examples are discussed to demonstrate the accuracy and robustness of the proposed formulation: the Taylor–Green vortex, three-dimensional vortex ring impinging on a wall at Re 570, fluid–structure interaction of falling plates and the flow around a sphere at $Re = 10^4$. Finally a summary and suggestions for future work is given in Section 6.

2. Problem formulation

In the present study we will consider dynamical systems consisting of fluid flow interacting with moving/deforming immersed bodies. The flow is always incompressible, and transitional/turbulent flow patters may be present. An example is

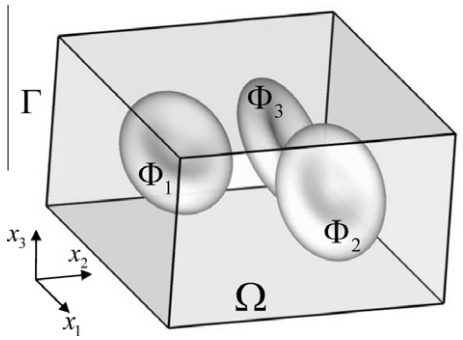

Fig. 1. An example of a dynamical system, where three solid bodies Φ_1, Φ_2 and Φ_3 interact with the flow in domain Ω, with boundary Γ.

shown in Fig. 1, where a set of solid bodies, Φ_1, Φ_2 and Φ_3, interacts with the flow within the domain Ω bounded by Γ. In this setting, the dynamics of the fluid and structures are described by different sets of equations, which need to be solved as a coupled system. On the side of the fluid the LES modeling framework is adopted, and the spatially-filtered Navier–Stokes equations for incompressible flow are solved:

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial \bar{u}_i \bar{u}_j}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} + \frac{1}{Re}\frac{\partial^2 \bar{u}_i}{\partial x_j \partial x_j} + f_i, \tag{1}$$

$$\frac{\partial \bar{u}_i}{\partial x_i} = 0, \tag{2}$$

where $x_i (i = 1, 2, 3)$ are the Cartesian coordinates, $\bar{u}_i$ are the resolved velocity components in the corresponding directions, $\bar{p}$ is the resolved pressure, and f_i represents an external body force field. The equations are non-dimensional and $Re = UL/\nu$ is the Reynolds number (U is a reference velocity, L a reference length scale and ν is the kinematic viscosity of the fluid). In LES the large scales are resolved directly as in a DNS, and all scales smaller than the filter size, which is usually proportional to the local grid size, are modeled. In Eq. (1) the effect of the unresolved scales appears in the sub-grid scale (SGS) stress term, $\tau_{ij} = \overline{u_i u_j} - \bar{u}_i \bar{u}_j$, which needs to be parameterized. In all LES reported in the present study the SGS stresses are modeled using the Lagrangian dynamic eddy-viscosity (LDEV) model [29]. Details on the present implementation can be found in [35].

The motion of a set of rigid bodies within the fluid domain, Ω, is governed by a set of ordinary differential equations (ODEs) of the form [8]:

$$\begin{bmatrix} [\mathbf{I}] & [\mathbf{0}] \\ [\mathbf{0}] & [\mathbf{M}(\mathbf{q}_1)] \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}}_1 \\ \dot{\mathbf{q}}_2 \end{Bmatrix} = \begin{Bmatrix} \mathbf{q}_2 \\ \mathbf{F}(\mathbf{q}_1, \mathbf{q}_2, t) \end{Bmatrix}, \tag{3}$$

where $\mathbf{q}_1 = [\, q_1 \quad q_2 \quad \dots \quad q_n \,]^T$ is a vector containing the set of n generalized coordinates of the structural system and $\mathbf{q}_2 = \dot{\mathbf{q}}_1$is the set of generalized velocities. [**I**]is the n x n identity matrix, [$\mathbf{M}(\mathbf{q}_1)$] is the nonlinear mass matrix and $\mathbf{F}(\mathbf{q}_1,\mathbf{q}_2, t)$ is a vector containing damping, rotation derived, elastic and externally applied forces. This last set of forces in our case, includes the gravity force and fluid tractions in the direction of the coordinates q_i ($i = 1,\dots,n$). Note that for deformable elastic bodies, the discretization of their governing equations will also lead to a system of the form given by Eq. (3). Therefore, the solution procedure presented here can be extended to deformable bodies in a straightforward manner.

3. Adaptive mesh refinement

3.1. Grid topology

The computational grid consists of a number of nested grid blocks with $nx \times ny \times nz$ computational cells at different refinement levels, Ω^l, $l = 0,1,\dots,l_{\max}$. The coarsest grid blocks at level Ω^0 always cover the entire computational domain, and local refinement is achieved by the bisection of selected blocks in every coordinate direction. In this process an arbitrary block, b, at level l, for example, will be the origin of eight *children* blocks at level $l + 1$ that occupy the same volume as their *parent* block. We call *leaf* blocks the blocks at the highest level of refinement present on a particular region of the domain (blocks that have not been refined, and therefore have no children). Any leaf-block at level l shares a common boundary with leaf-blocks whose refinement level may differ from l by at most one level. An example is shown in Fig. 2(a), where an S-shaped domain is discretized with an AMR grid. Level Ω_0 covers the entire domain and consists of 6 blocks (gray). The grid is locally refined near the center of the domain by adding two levels of refinement, Ω^1 (orange) and Ω^2 (yellow). As a result the cell size in the ith direction for a block at the finest level, $l = 2$, is only a quarter of the one at $l = 0 : \Delta x_i^2 = \Delta x_i^0/4$, where $i = 1, 2, 3$. Note that given the above constraints, Δx_i^l for computational cells on adjacent leaf-blocks at different refinement levels will differ by a factor of two. The resulting grid structure is managed using the octree data-structure in the Paramesh toolkit [25], which enables a robust implementation and straightforward parallelization of the proposed algorithm.

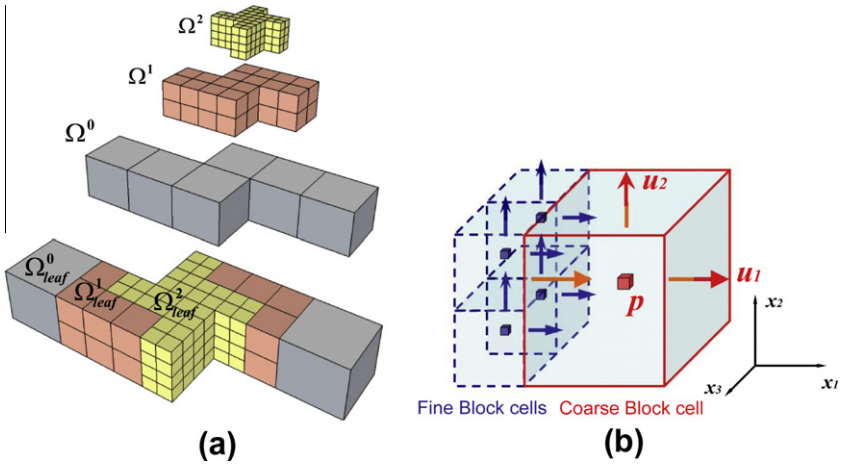

Fig. 2. (a) Example grid hierarchy and nested sub-blocks. Three refinement levels, Ω^0, Ω^1 and Ω^2 have been used. (b) Staggered grid arrangement in cells adjacent to a coarse–fine interface: pressure is located at the center and velocities at the face. The velocity component u_3 in the x_3 direction is not shown for clarity.

A staggered variable arrangement is used in each grid block as shown in Fig. 2(b), where the velocity component in the x_3 direction has been omitted for clarity. In the remainder of the paper we will drop the overbar indicating filtered variables, and denote the pressure and velocities for block b at level l as p^{bl}_{jkm} and $u^{bl}_{i,jkm}, i = 1,2,3$ respectively. jkm are indices that identify a grid cell within block b, and i refers to the orientation of the velocity component. Other variables such as the turbulent viscosity, v_t, can be similarly defined. We will also denote $P(b)$ as the parent block of block b, and $C(b,i)$ as the i^{th} child block of block b, where $i = 1,\ldots,2^d$ and d is the number of dimensions of the problem.

3.2. Prolongation and restriction operators

Let us now define the prolongation and restriction operators that are needed to transfer variables between parent and children blocks. The general form of a restriction operation of a variable ϕ from block $C(b,i)$ at level $l+1$ to the parent, $P(b)$, at level l is given by

$$(R\phi)^{P(b),l}_{ijk} = \sum_{p,q,r=il}^{el} \alpha'_{pqr}\phi^{b,l+1}_{i'+p,j'+q,k'+r}, \tag{4}$$

where, ijk are the indexes of a cell on $P(b)$ containing the restricted variable and $i'j'k'$ are the indexes of a cell in b. The limits il and el define the interpolation stencil, where the stencil size and interpolation coefficients, α'_{pqr}, depend on the interpolation scheme used. The prolongation of a variable from block $P(b)$ to b, on the other hand, is given by

$$(I\phi)^{b,l+1}_{i'j'k'} = \sum_{p,q,r=il}^{el} \alpha_{pqr}\phi^{P(b),l}_{i+p,j+q,k+r}, \tag{5}$$

On a staggered grid, prolongation and restriction operators are defined separately for the flow variables collocated at the cell-centers (p^{bl}_{jkm}, $v^{bl}_{t_{jkm}}$, etc.) and the cell faces ($u^{bl}_{i,jkm}$). For the former we use a dimension-by-dimension interpolation strategy that utilizes second-order Lagrange polynomials to perform one-dimensional sweeps over a three-dimensional stencil. In these cases the weight factors, α_{pqr} and α'_{pqr}, in Eqs. (4) and (5) are products of the corresponding coefficients in the one-dimensional Lagrange polynomials. For the face-centered components we exploit the fact that they are co-planar, and the above strategy is employed in two-dimensions. Example restriction operations in a two-dimensional staggered grid are shown in Fig. 3(a). The pressure, p^l_{ij}, located at cell-center, ij, of the parent block, is interpolated from pressures on a 3×3 stencil from the corresponding children blocks. The velocities u^l_{ij} and v^l_{ij}, however, are found using the one-dimensional stencils (two-dimensional in three-dimensions) shown in the figure. Note that in Fig. 3 the superscript indicating the block has been dropped for simplicity and the parent and children blocks are identified by their level superscripts, l and $l+1$ respectively.

The operators given by Eqs. (4) and (5) satisfy the accuracy requirements but do not guarantee the divergence-free evolution of the velocity field through the AMR grid. In the present implementation this property is particularly desirable whenever prolongations and restrictions are performed as part of the local grid adaptivity (local refinement and derefinement) from timestep to timestep. Divergence-free restriction operators can be constructed in a fairly straightforward manner. In our staggered grid arrangement, for example, using face-averaging for all the velocity components preserves the divergence of the velocity vector. Divergence-preserving prolongation of a vector field, on the other hand is not straightforward. Martin et al. [27], in their AMR implementation for the Navier–Stokes equations for incompressible flow, use a divergence cleaning

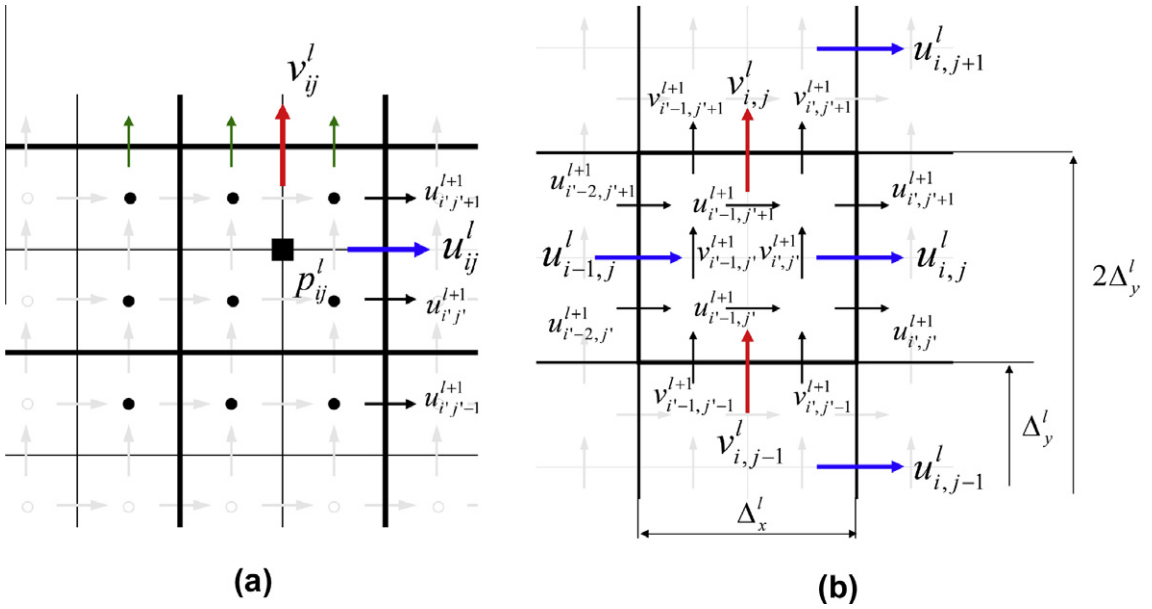

Fig. 3. (a) Interpolation stencil utilized by two-dimensional prolongation operators. It contains nine points for cell-centered variables such as, p^l_{ij}, and three points for variables located at the cell faces, such as u^l_{ij} and v^l_{ij}. (b) Variable arrangement for the construction of divergence-preserving prolongation operators in two-dimensions.

methodology employing an extra Poisson solution. Balsara [7] in the framework of his MHD solver, proposed a divergence-free prolongation operator applicable to AMR meshes with arbitrary refinement ratios.

In the present work we have developed divergence-preserving prolongation operators tailored to the specific AMR topology, where the grid size between consecutive refinement levels can only differ by a factor of two. We will describe the proposed scheme in two-dimensions for the configuration shown in Fig. 3(b). Let us define the discrete divergence, D, at an arbitrary cell ij of the parent block as

$$D\mathbf{u} = \frac{u^l_{i,j} - u^l_{i-1,j}}{\Delta^l_x} + \frac{v^l_{i,j} - v^l_{i,j-1}}{\Delta^l_y}, \tag{6}$$

where Δ^l_x and Δ^l_y is the grid spacing on the parent block in the x and y directions, respectively. The above is also the target divergence for the reconstructed velocity field on the corresponding children blocks. We will first determine the eight velocity components, $u^{l+1}_{i'-2,j'}, u^{l+1}_{i'-2,j'+1}, u^{l+1}_{i',j'}, u^{l+1}_{i',j'+1}, v^{l+1}_{i'-1,j'-1}, v^{l+1}_{i',j'-1}, v^{l+1}_{i'-1,j'+1}, v^{l+1}_{i',j'+1}$, which are located at the faces of the parent block as shown in Fig. 3(b). This is done using one-dimensional, quadratic, mass-flux preserving interpolations on each face. For example, on the right face of the coarse cell one can assume that the velocity can be approximated as, $u(y) = a_0 + a_1 y + a_2 y^2$. Then, utilizing the the known values, $u^l_{i,j-1}$, $u^l_{i,j}$ and $u^l_{i,j+1}$ on that face, together with the discrete mass-flux conservation constraint, $u^l_{i,j} = 0.5(u^{l+1}_{i',j'} + u^{l+1}_{i',j'+1})$, the following algebraic system can be assembled:

$$\begin{aligned}
u^l_{i,j-1} &= a_0 + a_1 \frac{\Delta^l_y}{2} + a_2 \left(\frac{\Delta^l_y}{2}\right)^2, \\
u^l_{i,j} &= \frac{1}{2}\left(u^{l+1}_{i',j'} + u^{l+1}_{i',j'+1}\right) = a_0 + a_1 \frac{3\Delta^l_y}{2} + a_2 \frac{37}{16}\left(\Delta^l_y\right)^2, \\
u^l_{i,j+1} &= a_0 + a_1 \frac{5\Delta^l_y}{2} + a_2 \left(\frac{5\Delta^l_y}{2}\right)^2.
\end{aligned} \tag{7}$$

The solution of (7) yields the coefficients α_0, α_1, α_2, and the unknown velocities at the midpoints of the face can the be found in a straightforward manner. For example, $u^{l+1}_{i',j'} = a_0 + 5a_1\Delta_y/4 + a_2(5\Delta_y/4)^2$. Note that the system (7) can be written in matrix form, and the inverse of the resulting 3×3 Vandermode matrix can be found analytically. The same approach is used on all faces of the parent cell.

Next, the remaining four interior velocities, $u^{l+1}_{i'-1,j'}, u^{l+1}_{i'-1,j'+1}, v^{l+1}_{i'-1,j'}$ and $v^{l+1}_{i',j'}$, shown in Fig. 3(b) are determined. For each of the children blocks one can write the discrete divergence equations and set it equal to that from (6). For example, for the block $(i' - 1, j' + 1)$ in Fig. 3(b):

$$\frac{u^{l+1}_{i'-1,j'} - u^{l+1}_{i'-2,j'}}{\Delta^{l+1}_x} + \frac{v^{l+1}_{i'-1,j'} - v^{l+1}_{i'-1,j'-1}}{\Delta^{l+1}_y} = D\mathbf{u}, \tag{8}$$

where Δ^{l+1}_x and Δ^{l+1}_y is the grid spacing on the children block in the x and y directions respectively. Using similar expressions for all four blocks we can assemble a system of four equations with four unknowns. It is of the form, $\mathbf{A}\mathbf{u}_\mathbf{m} = \mathbf{b}$, where $\mathbf{A}$ the 4×4 matrix and $\mathbf{u}_m$, $\mathbf{b}$ the unknowns and source vectors respectively. $\mathbf{A}$, however, is a matrix of rank three and cannot be inverted. To address this issue we will consider an additional independent equation, i.e. express $u^{l+1}_{i'-1,j'}$ (or any of the internal velocities) using quadratic interpolation:

$$u^{l+1}_{i'-1,j'} = -\frac{1}{8}u^{l+1}_{i'-4,j'} + \frac{3}{4}u^{l+1}_{i'-2,j'} + \frac{3}{8}u^{l+1}_{i',j'}. \tag{9}$$

The resulting system of equations can be written as

$$\begin{bmatrix} 1 & 0 & \delta & 0 \\ -1 & 0 & 0 & \delta \\ 0 & -1 & 0 & -\delta \\ 0 & 1 & -\delta & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} u^{l+1}_{i'-1,j'} \\ u^{l+1}_{i'-1,j'+1} \\ v^{l+1}_{i'-1,j'} \\ v^{l+1}_{i',j'} \end{Bmatrix} = \begin{Bmatrix} \beta + u^{l+1}_{i'-2,j'} + \delta v^{l+1}_{i'-1,j'-1} \\ \beta + \delta v^{l+1}_{i',j'-1} - u^{l+1}_{i',j'} \\ \beta - u^{l+1}_{i',j'+1} + \delta v^{l+1}_{i',j'+1} \\ \beta + u^{l+1}_{i'-2,j'+1} - \delta v^{l+1}_{i'-1,j'+1} \\ \left(-u^{l+1}_{i'-4,j'} + 6u^{l+1}_{i'-2,j'} + 3u^{l+1}_{i',j'}\right)/8 \end{Bmatrix}, \tag{10}$$

where $\delta = \Delta^{l+1}_x / \Delta^{l+1}_y$, $\beta = \Delta^{l+1}_x D\mathbf{u}$. This is now an extended system of five equations and four unknowns which can be written as $\mathbf{u}_\mathbf{m} = (\mathbf{A}_\mathbf{e}{}^T\mathbf{A}_\mathbf{e})^{-1}\mathbf{A}_\mathbf{e}{}^T\mathbf{b}_\mathbf{e}$. The product $(\mathbf{A}^T_\mathbf{e}\mathbf{A}_\mathbf{e})^{-1}\mathbf{A}^T_\mathbf{e}$ is the 4×5 left-pseudo-inverse of the system, which can be found analytically. The addition of Eq. (9) changes the rank of the system and the above *best fit* solution is actually the only solution to the system. The extension of the above procedure to three-dimensions is straightforward, and the detailed equations are given in Appendix A.

We should also note that the above formulation does not reduce to the scheme proposed in [7] for a refinement ratio of two. In the latter, a piecewise-linear variation is assumed for the divergence-free vector field across each face of the rectangular region of interest (the coarse-grid cell in our case). The slopes on this variation are defined via the minmod slope-limiter. The vector field is then interpolated at all internal fine-grid locations using multidimensional quadratic polynomials, which are build with specific constraints. In the scheme described above we assume a parabolic velocity distribution that conserves the mass-flux across the interface, and is defined using neighboring coarse-grid velocities. Although we have not conducted direct comparisons between the two approaches, we anticipate that the proposed approach results in a reduction of the computational cost and memory requirements, of course, at the expense of the generality, since it is only applicable to interfaces with a refinement ratio of two.

3.3. Treatment of the block boundaries

To facilitate the discretization of the equations of motion at block boundaries, overlapping between neighboring blocks is created by means of two layers of ghost cells. A two-dimensional configuration for neighboring blocks at refinement levels l and $l+1$ is shown in Fig. 4. To assign values to the ghost cells on the grid block at level l: (i) the solution at level $l+1$ is restricted to its parent grid, which has the same level of refinement as the adjacent coarse block, using Eq. (4); (ii) *filling* of the ghost cells is done by simple 'ejection' of the corresponding variable from the parent grid. To assign values at ghost cells on the grid block at level $l+1$ a similar procedure is used, which utilizes the prolongation operator given by Eq. (5). In Fig. 4 the grid nodes in the interpolation stencil are shown for both cases.

3.4. Temporal integration scheme

A standard, second-order, fractional-step method is utilized for the temporal integration of the governing equations [23]. In our implementation both advective and diffusive terms are advanced in time using an explicit Adams–Bashforth scheme. On each leaf-block at level $l = 0, \ldots, l_{\max}$ the predicted velocity, $\tilde{u}^l_i$, can be written as

$$\tilde{u}^l_i = u^{l,n}_i + \frac{\Delta t}{2}\left(3H\left(u^{l,n}_i\right) - H\left(u^{l,n-1}_i\right)\right) - \Delta t\frac{\partial p^{l,n}}{\partial x_i} + \Delta t f^{n+1/2}_i, \tag{11}$$

where H is a discrete operator containing the convective, viscous and SGS terms, the superscript, n, refers to the time level, and Δt is the timestep, which is the same on all refinement levels, l. All spatial derivatives are approximated using second-order central differences on a staggered grid. The predicted velocity field, $\tilde{u}^l_i$, which is not divergence free, can be projected into a divergence-free space by applying a correction of the form:

$$u^{l,n+1}_i = \tilde{u}^l_i - \Delta t\frac{\partial}{\partial x_i}(\delta p^l), \tag{12}$$

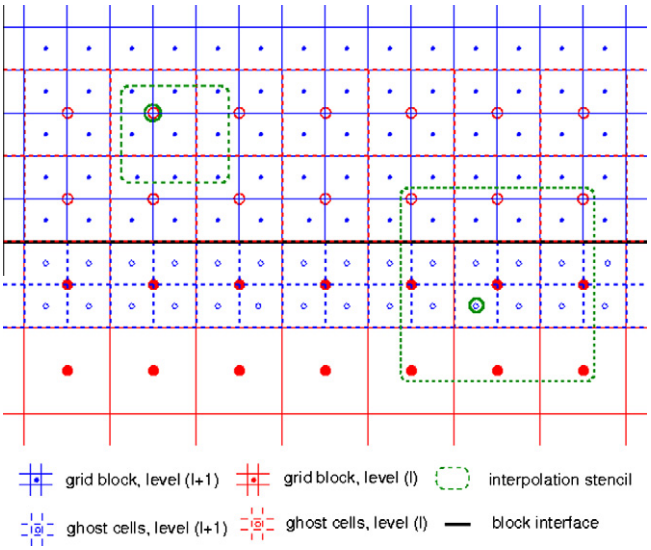

Fig. 4. Ghost cell configuration at the interface between blocks at levels l and $l+1$. Filled symbols denote the centers of interior cells and open symbols the centers of the ghost cells. The grid nodes surrounded by the green dotted lines are the ones on the interpolation stencil for the ghost cells identified by a green circle. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where $\delta p^l = p^{l,n+1} - p^{l,n}$ is the pressure correction, which satisfies the following Poisson equation:

$$\frac{\partial^2(\delta p^l)}{\partial x_i \partial x_i} = \frac{1}{\Delta t}\frac{\partial \widetilde{u}_i^l}{\partial x_i}. \tag{13}$$

The solution of the Poisson equation (13), is done using the multigrid algorithm developed by Martin and Cartwright [26]. This method has been designed for block-structured adaptive grids and maintains second-order spatial accuracy, imposing continuity of δp^l and its derivatives across interface jumps. It uses a residual-correction formulation and employs red–black Gauss Seidel (RBGS) point relaxation at each level. In our implementation, in order to increase the performance of the solver, we exploit the uniformity of the mesh at the coarsest level of the V-cycle, l_{solve}, which covers the entire computational domain. In particular, we utilize a direct solver, rather than RBGS iterations, to solve the residual-correction equation at that level. The only drawback of this strategy is that the computational domain has to be rectangular since the direct solver utilizes FFT or Cosine transforms, depending on the boundary conditions. This strategy was found to reduce the number of levels on a V-cycle and relaxation-communication operations, while maintaining the convergence rate of the multigrid scheme.

An important issue with the application of projection schemes to S-AMR grids is related to conservation of mass at coarse–fine block interfaces. In Fig. 5 a two-dimensional example of the interface between cells at levels l and $l+1$ is shown. The velocity component, u_{ij}^l, normal to the interface at the coarse level l, as well as the corresponding velocity components, $u_{i'j'}^{l+1}$ and $u_{i'j'+1}^{l+1}$, at fine level $l+1$, are unknowns to be determined during the solution process. They need, however, to satisfy an additional constraint coming form the conservation of the mass-flux across the interface:

$$u_{ij}^l \Delta y^l = u_{i'j'}^{l+1}\Delta y^{l+1} + u_{i'j'+1}^{l+1}\Delta y^{l+1}, \tag{14}$$

where Δy^l is the grid spacing at level l and $\Delta y^{l+1}=\Delta y^l/2$, is the grid spacing at level $l+1$. In general, Eq. (14) will not hold after the computation of the velocities normal to the refinement jump, resulting to a flux mismatch localized in grid refinement interfaces. To alleviate this problem one can assume that the velocity components on the fine grid at level, $l+1$, are more accurate than the corresponding coarse-grid component, u_{ij}^l, which can then be corrected to satisfy Eq. (14) (see for example: [13,2]). In the present formulation the flux correction is realized in the following manner: (i) the intermediate velocities at the fine level, $\widetilde{u}_{i'j'}^{l+1}$ are computed from Eq. (11); (ii) the intermediate velocity on the coarse level, $\widetilde{u}_{ij}^l$, is then corrected using Eq. (14); (iii) during the iterative solution of the Poisson equation (13) the gradient of δp normal to the refinement interface at the u_{ij}^l location is forced to satisfy the following equation:

$$\left.\frac{\Delta(\delta p^l)}{\Delta x}\right|_{ij}\Delta y^l = \left.\frac{\Delta(\delta p^{l+1})}{\Delta x}\right|_{i'j'}\Delta y^{l+1} + \left.\frac{\Delta(\delta p^{l+1})}{\Delta x}\right|_{i'j'+1}\Delta y^{l+1}; \tag{15}$$

(iv) it is now trivial to show that the corrected velocity from Eq. (12) will also conserve the mass-flux across the interface.

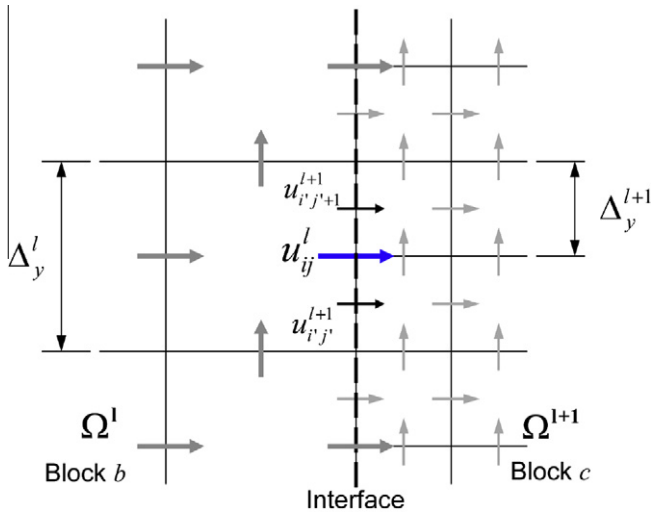

Fig. 5. Velocity components contributing to the mass-flux across a coarse–fine interface.

4. Treatment of complex moving boundaries

4.1. Direct-forcing scheme

To compute the flow around complex bodies immersed in the nested structured grids an embedded-boundary, direct-forcing approach is utilized. In such case the requirement for the grid to conform to the body is relaxed (see Fig. 6(a)), and boundary conditions are imposed using a forcing function constructed in discrete space. The motion of the immersed body can be either prescribed or it is the product of the interaction with the flow. The main feature of our formulation is that the forcing function is computed directly on the immersed body, which is represented by a series of Lagrangian markers, rather than the neighboring grid nodes on the structured grid, as it is done in classical direct-forcing methods (i.e. [18,5]).

In particular, we first take a provisional step to compute the intermediate velocity, $\hat{u}_i$, as if no immersed bodies are present in the flow domain. In practice we use Eq. (11) with f_i set to zero. The resulting velocity, $\hat{u}_i$, will not satisfy the incompressibility constraint and the boundary conditions on the immersed body. Note that in the discussion below we will drop the grid level superscript, l, from all variables, since all operations involving immersed bodies are conducted within the same block. Next we build the transfer operators, that will enable transfer of information from the Lagrangian to the Eulerian grid and vise-versa, using moving least squares (MLS) shape functions with compact support [41]. To facilitate this process, we identify the closest Eulerian grid node to each Lagrangian marker. Referring to Fig. 6(b), for example, the marker L_a is asso-

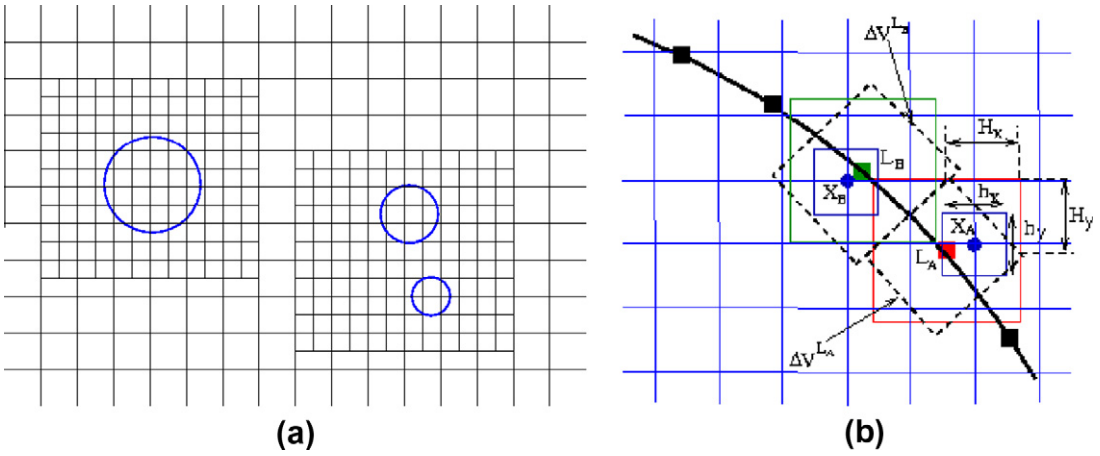

Fig. 6. (a) Sketch of the locally refined computational grid around multiple bodies. (b) Definition of the MLS support-domain for two neighboring Lagrangian markers, L_A and L_B, which are color coded for clarity. X_A and X_B denote the closest Eulerian nodes to L_A and L_B, respectively. The corresponding volumes ΔV are also shown (dashed line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

ciated to the grid node x_a, which is in the center of a cell with dimensions h_x and h_y in the x and y directions respectively. Marker L_b is associated to the grid node x_b and so on. Note that more than one Lagrangian markers from the same, or different immersed bodies, can be associated with the same Eulerian grid node. We then define a support-domain around each Lagrangian marker, in which the shape functions are constructed. In our case the support-domain is a rectangular box of size $2H_x \times 2H_y \times 2H_z$ centered at the location of the marker. H_x, H_y and H_z are different for each marker and are proportional to the local Eulerian grid. After building the shape functions we can then find the fluid velocity, $\widehat{U}_i^L$, on the location of each Lagrangian marker by interpolating from $\hat{u}_i$:

$$\widehat{U}_i^L(\mathbf{x}) = \sum_{k=1}^{ne} \phi_k^L(\mathbf{x}) \hat{u}_i^k, \tag{16}$$

where capital letters refer to Lagrangian quantities, ne is the total number of points on the interpolation stencil, $\phi_k^L(\mathbf{x})$ are the shape functions for marker point L. If U_i^B is the velocity one wishes to enforce on the Lagrangian marker, L, then the direct-forcing function on this marker can be computed from, $F_i^L = (U_i^B - \widehat{U}_i)/\Delta t$ [39]. The forcing function, f_i, is then obtained by transferring F_i^L back to the Eulerian grid as follows:

$$f_i^k = \sum_{L=1}^{nl} c_L \phi_k^L F_i^L, \tag{17}$$

where f_i^k is the volume force in the Eulerian point k in the direction i, and nl is the number of Lagrangian markers which have been related to the grid point k. The coefficient c_L is used to rescale the shape functions in a way that the total force acting on the fluid is not changed by the transfer. The forcing function, f_i is then introduced in Eq. (11) to compute the predicted velocity, $\tilde{u}_i^l$, which now satisfies the boundary conditions on the immersed body. The overall formulation utilizes very compact stencils and, without compromising accuracy and robustness, gives results that are identical to 'sharp' direct-forcing methods. Details, as well as an extensive validation can be found in [41]. We should also note that the AMR formulation outlined in the previous sections can be coupled to any direct-forcing scheme (i.e. [18,5]) in a straightforward manner.

4.2. Fluid–structure interaction algorithm

A fundamental complication with two-way, fluid–structure interaction (FSI) problems, is that the prediction of the flow and the corresponding hydrodynamic loads requires knowledge of the motion of the structure and vice versa. In the present study a strong-coupling scheme is adopted, where the fluid and the structure are treated as elements of a single dynamical system, and all of the governing equations are integrated simultaneously and iteratively in the time domain. The method is based on Hamming's 4th-order, predictor–corrector formulation, which avoids the evaluation of the hydrodynamic loads at fractional timesteps. The details of the overall approach and a demonstration of the accuracy and efficiency for a variety of fluid–structure interaction problems in viscous incompressible flows can be found in [43]. In Fig. 7 a flowchart of the overall algorithm as adapted to our AMR implementation is shown:

(i) At the beginning of each timestep, and only for the case of LES, the SGS eddy viscosity is computed using the LDEV model. We should note that the use of eddy resolving methods, such as LES, within the AMR framework is not straightforward. This is due to the fact that in LES the flow field is generally not smooth at the smallest scale (the filter width is not much larger than the grid size), so that numerical errors in the interpolation between grids can be significant. In addition, the SGS eddy viscosity used to represent the effect of the unresolved scales is usually proportional to the filter width (and, in most cases, to the grid size) squared. A sudden mesh refinement or coarsening may result in a discontinuity in eddy viscosity, which can generate significant errors. In the present implementation we utilize smoothly varying filter-widths at the fine-coarse interfaces between blocks, together with explicit filtering of the advective term. We found that this strategy results in smoother *transitions* between blocks at different refinement levels eliminating unphysical jumps in the resolved turbulent kinetic energy. Details can be found in [40].

(ii) Compute the provisional velocity, $\hat{u}_i^l$, which does not satisfy boundary conditions on the immersed boundary, and is not divergence free. Assign values for $\hat{u}_i^l$ to the ghost cells as described in Section 3.3.

(iii) Perform AMR if necessary (every n steps). In particular, all leaf-blocks are examined and flagged for refinement or derefinement, according to specified criteria. For example, in all FSI cases reported below, a leaf-block is selected for refinement when it contains an immersed body, or when the vorticity magnitude is larger than a predetermined threshold. A leaf-block is marked for derefinement, on the other hand, when an immersed body is not present within the block, and the vorticity is below a threshold. Other criteria, such as velocity error norms, SGS dissipation etc., could also be used for this purpose. In the case of refinement, the newly created children blocks are added to the octree. All variables other than $\hat{u}_i^l$ and the discrete operator $H(u_i^{l,n-1})$ in (11), are interpolated using Eq. (5). For $\hat{u}_i^l$ and $H(u_i^{l,n-1})$, the divergence-preserving prolongation procedure discussed in Section 3.2 is used. For each leaf-block flagged for derefinement, we first check if its parent block contains children that are all flagged for derefinement. In such case we perform a restriction step and remove the corresponding leaf-blocks from the octree. Quadratic restriction using Eq. (4) is utilized for all variables except $\hat{u}_i^l$ and $H(u_i^{l,n-1})$, where linear restriction is used. The latter is a divergence-preserving operation.

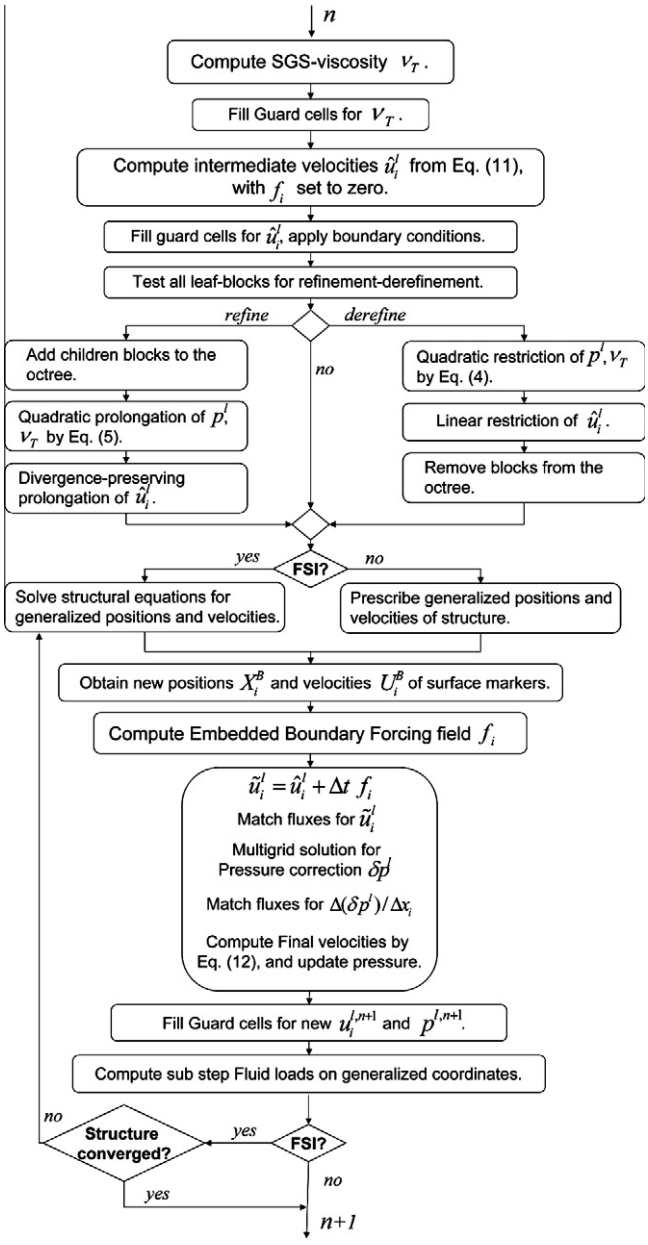

Fig. 7. Summary of fluid–structure interaction strategy.

(iv) In the case of FSI we apply a set of predictor–corrector sub-iterations as shown in the figure. Convergence is assumed when the L_2 error on structures generalized coordinates and velocities is less than a certain tolerance. In all FSI computations in the present study the tolerance is set to 10^{-8}.

5. Results

In this section we present a series of test problems with increasing complexity to demonstrate the accuracy and robustness of the proposed AMR formulation. Initially the spatial and temporal accuracy of the method is demonstrated for the Taylor–Green vortex problem. Then the case of a three-dimensional vortex ring impinging on a wall is considered. Finally two cases, where the accuracy and efficiency of the method in the presence of immersed bodies is examined, are presented: the FSI problem of two falling plates, and LES of the flow around a sphere at $Re = 10^4$. We should note that formal accuracy of the direct-forcing scheme utilized in the last two cases as well as the strong coupling FSI approach used in the case of the falling plates has been demonstrated for a single-block configuration in [41,43], respectively.

5.1. Taylor–Green vortex

To investigate the numerical accuracy of the method the Taylor–Green vortex problem is considered. The flow field is represented by an array of periodic counter-rotating vortices that decay in time, and has an analytical solution of the form:

$$u_a = -e^{-2\nu t}\cos x \sin y, \tag{18}$$

$$v_a = e^{-2\nu t}\sin x \cos y, \tag{19}$$

$$p_a = -\frac{e^{-4\nu t}}{4}(\cos 2x + \cos 2y), \tag{20}$$

where u_a and v_a are the velocity components in the x and y directions, respectively, p_a is the pressure, ν is the kinematic viscosity, and t is the time. The size of our computational domain was $2\pi \times 2\pi$, requiring the use of a mix of homogeneous Dirichlet and Neumann boundary conditions for the velocity components. Grids with two levels of refinement, as well as uniform ones are considered. In the latter case 32^2, 64^2, 128^2 and 256^2 grid points are used and the ghost-cell filling is performed by simple injection of the data between blocks sharing a common boundary. In the former case the base grids utilize 16^2, 32^2, 64^2 and 128^2 computational points on their coarse levels, and the refinement is performed in the second and fourth quadrants of the domain (see Fig. 8). To investigate the impact of the ghost-cell filling scheme on the overall accuracy of the method both linear and quadratic interpolation schemes are considered. In all cases the equations of motion are integrated for a total of $T = 0.03$ time units using a timestep of $\Delta t = 5.0 \times 10^{-5}$. The L_2 norm of the residual for the Poisson solution was kept in the order of 10^{-14}.

In Fig. 8 pressure isolines at $t = 0.03$ are shown for the AMR grid with two levels of refinement (level 0 being 32^2). It is evident that the main features of the flow are captured. In Fig. 9 the L_{inf} error norm for both velocity components at $t = 0.03$, is shown for all cases as a function of the grid size, Δ. Note that in all the two-level calculations, Δ, is computed from the highest refinement level. As expected, in all uniform grid cases second-order accuracy is observed. In the case of AMR the use of linear interpolation for the ghost-cell filling reduces the spatial accuracy, which is now closer to first order. The higher order interpolation scheme described in Section 3.3, on the other hand, maintains the second-order for both velocity components. We should also note that the error in the case of the AMR grid is always higher when compared to that of a uniform grid with equivalent resolution. This is due to the fact that the error in this problem is uniformly distributed in the computational domain and is proportional to the local velocity magnitude. As a result the $L_{\inf}$ error norm in the part of the computational box with highest resolution in the AMR case, will be affected by the neighboring coarser level, due to the inherent ellipticity of the problem. The difference in the L_2 error norm, on the other hand, (not shown in Fig. 9) between uniform and AMR grids is much smaller.

In all the above computations the grid is locally refined, but does not evolve with time, as required in most moving boundary problems. To examine the effects of the dynamic adaptation of the grid on the accuracy of the solver we also utilize

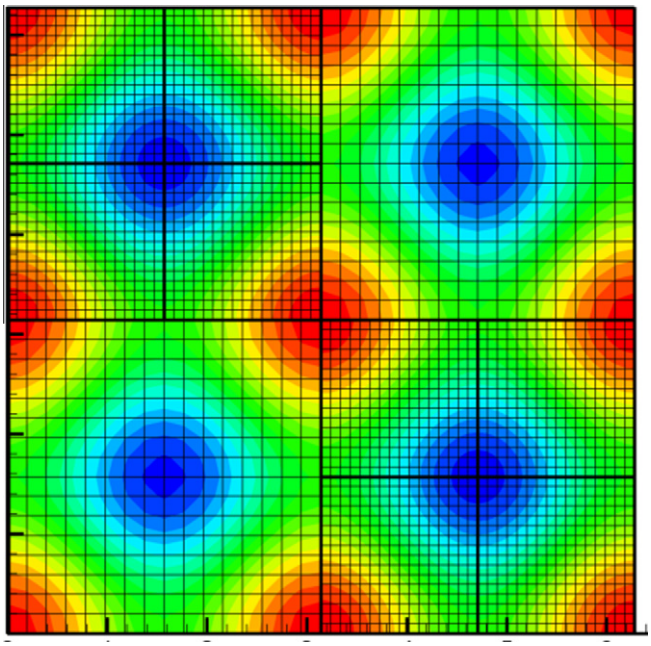

Fig. 8. AMR grid configuration with two refinement levels for the case of the Taylor–Green vortex. Pressure isolines ranging from $p = -0.4$ to 0.4 at $t = 0.03$ are also shown.

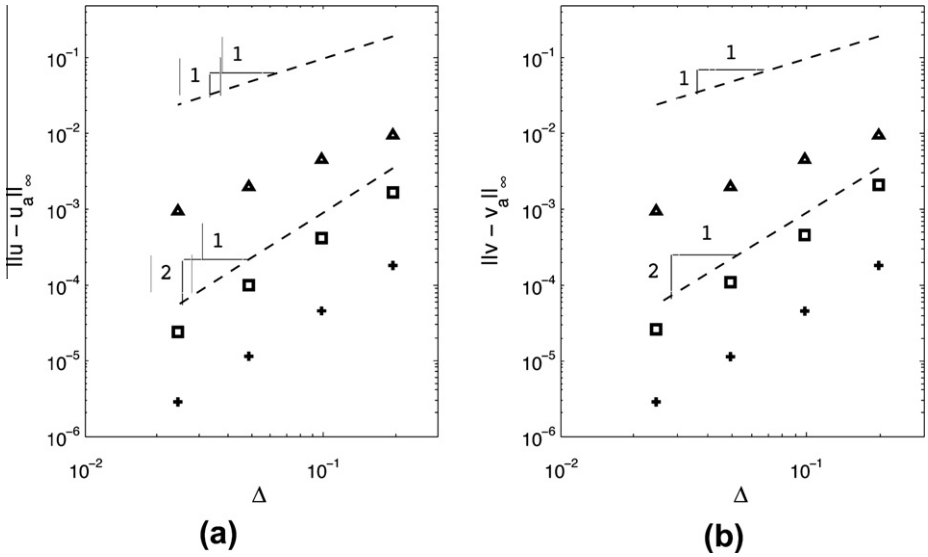

Fig. 9. Accuracy study on the Taylor–Green vortex problem. (a) $\|u - u_a\|_\infty$; (b) $\|v - v_a\|_\infty$ as a function of grid spacing at $t = 0.03$. + uniform grid; △ AMR grid, linear interpolation; □ AMR grid, quadratic interpolation. Note that the 1st and 2nd order slopes have been added for clarity.

the Taylor–Green vortex problem with the same setup as above. In this case, however, starting from a single level grid we refine every 10 iterations by adding one refinement level (Fig. 8), and then derefine every 10 iterations. Refinement, for example, is performed at iteration count $n = 10, 30, 50, \ldots, 90$, and derefinement at $n = 20, 40, \ldots, 80$. At iteration number 100 the solution is compared to the analytical one, and the corresponding error norms for the velocity and pressure fields are computed. An important part of the dynamic grid adaptation is the prolongation of the solution at the newly generated children blocks in refinement areas, and/or the restriction of the solution from eliminated leaf-blocks to its parent, in derefinement areas. As we discussed in Section 4.2 we perform AMR after the computation of the provisional velocity, $\hat{u}_i^l$. One can show that $\hat{u}_i^l$ is within $O(\Delta t^2)$ of a divergence-free field (see for example [23]) and, therefore, its prolongation/restriction after a grid adaptation step should maintain this property, so the overall temporal accuracy is not affected. As we already discussed in Section 3.2, not all prolongation/restriction operators are divergence preserving.

To illuminate their effects on the accuracy of the solution we conducted computations with three different prolongation operators for the velocity field: linear and quadratic, which do not preserve the divergence of $\hat{u}_i^l$, and the quadratic divergence-preserving operator discussed in Section 3.2. In Fig. 10(a) the L_{inf} error norms for the velocities and pressure are shown as a function of the grid size for all cases. It is evident that the spatial accuracy is not affected by the choice of prolongation operator, and in all cases second-order accuracy maintained. As expected, the error using linear prolongation is about an order of magnitude higher than using the quadratic interpolation variants. The evolution of the error norms with respect to time, on the other hand, reveals some very interesting patterns, as seen in Fig. 10(b). The prolongation operators, which do not preserve the divergence of $\hat{u}_i^l$, result in a monotonically increasing error for both velocity components, while for the divergence-preserving prolongation operator the error is always $O(10^{-4})$. These effects are more profound in the behavior of the pressure, where the linear and quadratic prolongations result in an error jump of at least three orders of magnitude just after each refinement step, compared to the divergence-preserving scheme. The pressure recovers in a few timesteps, but overall the error grows over time. In the case of fluid–structure interaction problems these pressure oscillations can introduce spurious loading on the structure leading to large errors on the solution. For the divergence-preserving scheme the error jump is very small, indicating that the spurious oscillations are practically eliminated. It is important to note that, the choice of other projection scheme variants, where the pressure is not treated incrementally, or making use of high-resolution approximate projections [3] might have a beneficial effect in damping the observed pressure oscillations and numerical errors due to non-divergence-preserving interpolations.

5.2. Vortex ring impinging on a wall

To demonstrate the ability of the proposed AMR formulation to accurately represent vorticity dynamics, the simple yet very challenging problem of a vortex ring impinging on a wall, is considered. Due to the importance of wall-vortex interactions in many technological applications there is variety of reference data in the literature [42,17,31,36]. In the present work we will consider a setup analogous to the one reported in [36], where a vortex ring is generated in the center of the $x - y$ plane, and at a distance $z_o = 1.5D_o$ from the wall as shown in Fig. 11. The vortex ring was generated by introducing an impulsive body force of the form:

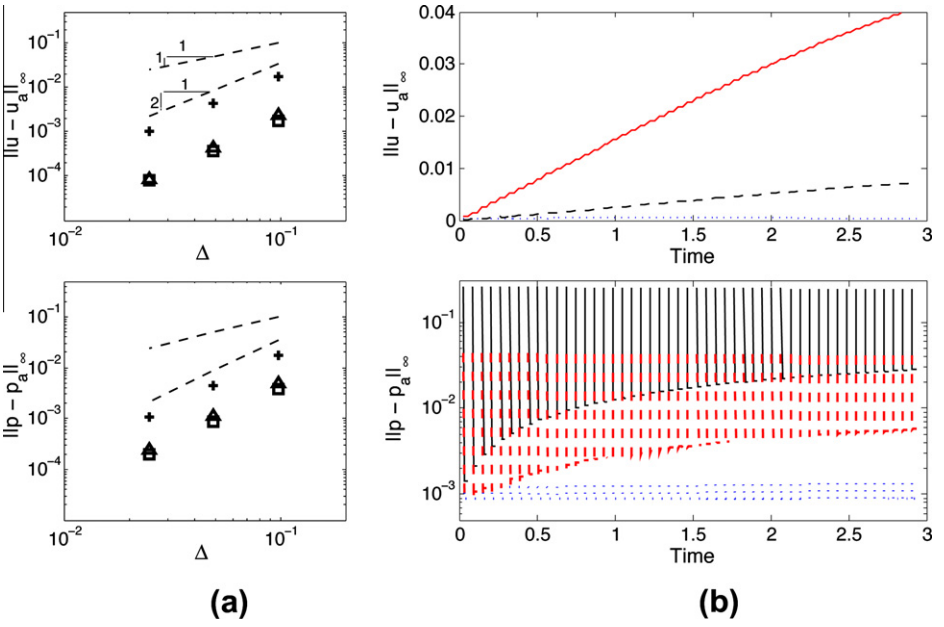

Fig. 10. Accuracy study on the Taylor–Green vortex problem where dynamic refinement/derefinement is performed every 10 timesteps. (a) $\|u-u_a\|_\infty$ (top), and $\|p-p_a\|_\infty$ (bottom) as a function of grid spacing at $t = 0.3$, + linear prolongation; △ quadratic prolongation; □ divergence-preserving prolongation; (b) $\|u-u_a\|_\infty$ (top), and $\|p-p_a\|_\infty$ (bottom) as a function of time — (black) linear prolongation; - (red) quadratic prolongation; ··· (blue) divergence-preserving prolongation. Note that the 1st and 2nd order slopes have been added for clarity. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

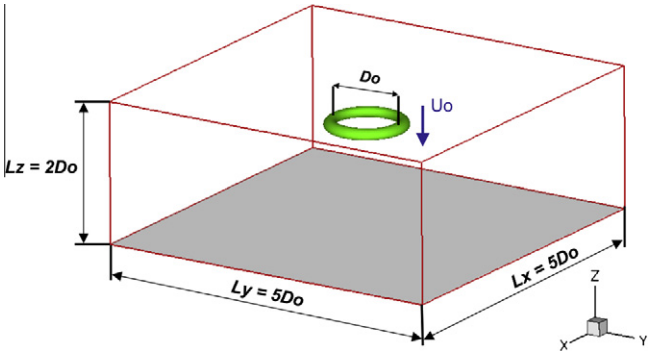

Fig. 11. Computational setup for the vortex ring impinging on a wall.

$$f_z(r,z,t) = -A_o T(t) F(z) H(r), \quad \text{where}$$
$$T(t) = 0.5[1+\tanh\alpha(\tau - t')], \quad F(z) = 0.5[1+\tanh\beta(B_z - z')], \quad H(r) = 0.5[1+\tanh\gamma(C_r - r)]. \tag{21}$$

Note that $t' = |t - t_o|$ and $z' = |z - z_o|$. Setting $A_o = 350$, $\alpha = 500$, $t_o = 0.05$, $\tau = 0.04$,$\beta = 100$, $B_z = 0.1$, $\gamma = 100$ and $C_r = 0.5$, the resulting vortex ring had $Re = U_o D_o/\nu = 570$, where U_o and D_o are its initial diameter and self-induced translation velocity and ν is the kinematic viscosity of the fluid. This is lower than the value of $Re_o = 645$, reported in [36]. A closer match of the initial conditions was not possible due to the fact that some of their forcing parameters were not listed.

The AMR grid was adaptively refined in areas of high velocity gradients. Five levels of refinement were used and the total number of points was of the order of 2×10^6. The grid adaptation was done every 10 timesteps using the modulus of the vorticity field, $|\boldsymbol{\omega}|$, as a test variable. The grid was refined in leaf-blocks where $|\boldsymbol{\omega}| > 5.5U_o/L_o$ anywhere within the block, and derefined if $|\boldsymbol{\omega}| < 4.0U_o/L_o$. Periodic boundary conditions are applied in the x and y directions, and non slip at the top and bottom walls in the z direction (see Fig. 11). The AMR solution is compared to a computation of exactly the same problem using a single-block, finite-difference solver [5]. The single-block grid in the latter case utilizes $256 \times 256 \times 128$ grid cells, and is uniform in the x and y directions, while it is stretched in the wall-normal direction, z. The resulting cell size in the near wall area was comparable to the one at the highest refinement level in the AMR computation.

In Fig. 12 vorticity contours are shown for both the single-block and AMR calculations. The vorticity normal to the $y-z$ plane is shown at four different times during the calculation. In the AMR case the block boundaries are also shown in the

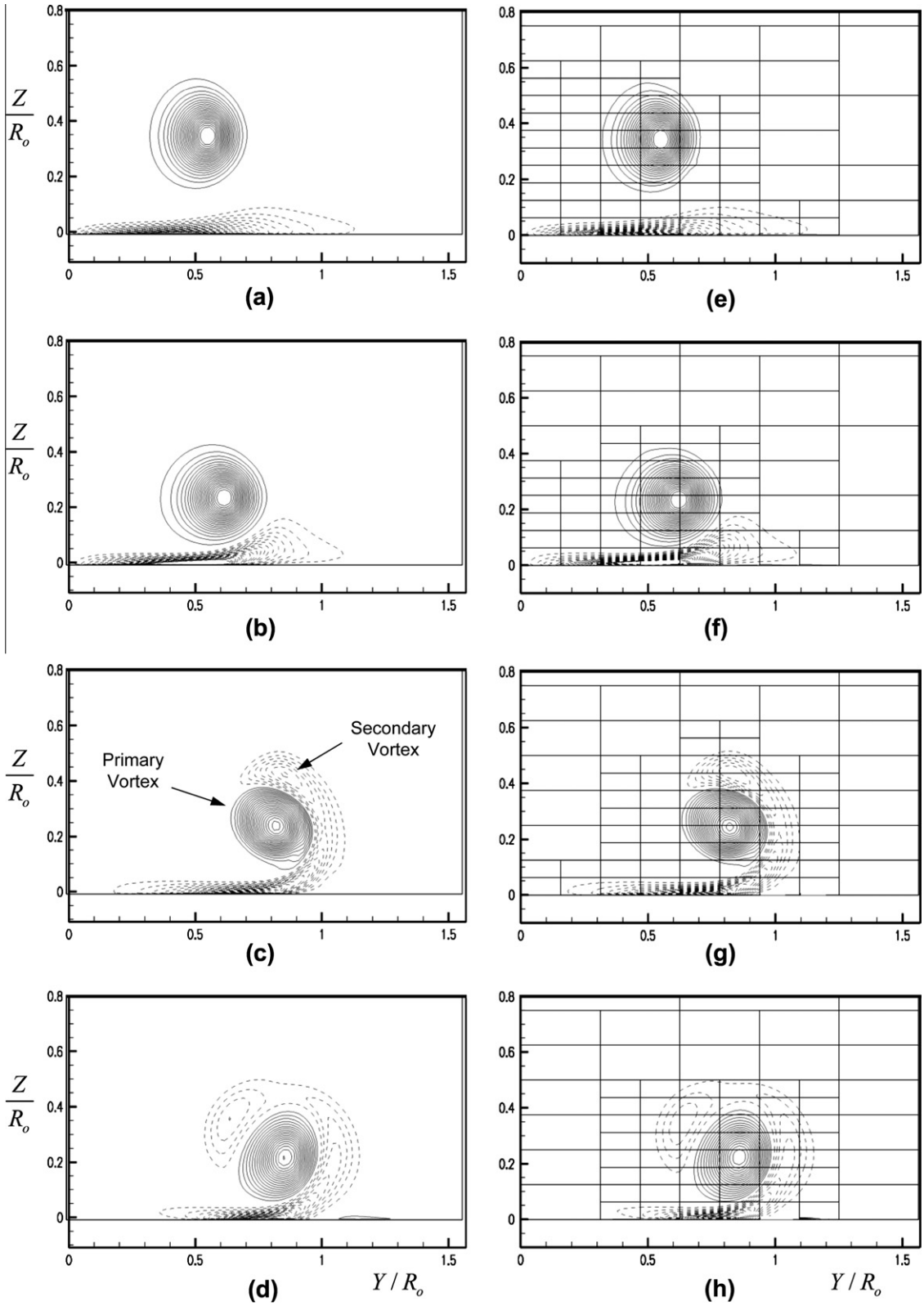

Fig. 12. Vorticity isolines at an $y - z$ plane for the case of the vortex ring impinging to a wall. Forty-five $\omega_x R_o/U_o$ contours from -50 to 50 are used. Left side is from the single-block calculation, and the right side from the AMR. (a) and (e) $t = 1.3$; (b) and (f) $t = 1.5$; (c) and (g) $t = 2.1$; (d) and (h) $t = 2.5$.

figure. The similarity between the two computations is striking. As the primary vortex approaches the wall there is an increase of its radius, which is accompanied by a noticeable decrease in the core size (i.e. Fig. 12(a)–(c). The boundary layer generated underneath the primary vortex thickens due to the adverse pressure gradient in the radial direction (Fig. 12(b)–(f)), and eventually the accumulated vorticity pinches-off forming a secondary vortex (Fig. 12(c)–(g)).

A three-dimensional view of primary and secondary vortex structures can be seen in Fig. 13, where isosurfaces of the second invariant of the velocity gradient tensor, $Q = -1/2(\partial\bar{u}_i/\partial x_j \partial\bar{u}_j/\partial x_i)$, are plotted. The secondary vortex orbits the primary vortex (Fig. 13(a)), and later undergoes a 'buckling' instability as it is compressed by the primary vortex (Fig. 13(b)). The secondary vortex breaks into smaller structures that are advected under the primary vortex and affect the vorticity generation near the wall. This results in small slender structures dominated by radial vorticity (see Fig. 13c). At the same time a third vortex pinches-off and also starts orbiting the primary one. This structure at this Re_o is fairly stable, and stays in the proximity of the primary vortex for the rest of the calculation (Fig. 13c).

In Fig. 14 the trajectories of the primary and secondary vortex centers are shown for both computations. The numerical results from Swearingen et al. [36] for a similar setup at $Re_o = 645$ are also included for comparison. The agreement between the single-block and AMR calculations is excellent and the trajectories of both vortices are identical indicating that the proposed AMR strategy properly captures the vorticity dynamics. The agreement with the results reported in [36] is also good. Small discrepancies near the wall are primarily due to the ambiguity in the definition of the vortex center as well as the differences in the initial conditions.

5.3. Fluid–structure interaction of two falling plates

To test the accuracy of the approach in fluid–structure interaction problems the case of two falling plates is considered (see Fig. 15). In this case, Eq. (3) governing the dynamics of each plate can be reduced to the following form:

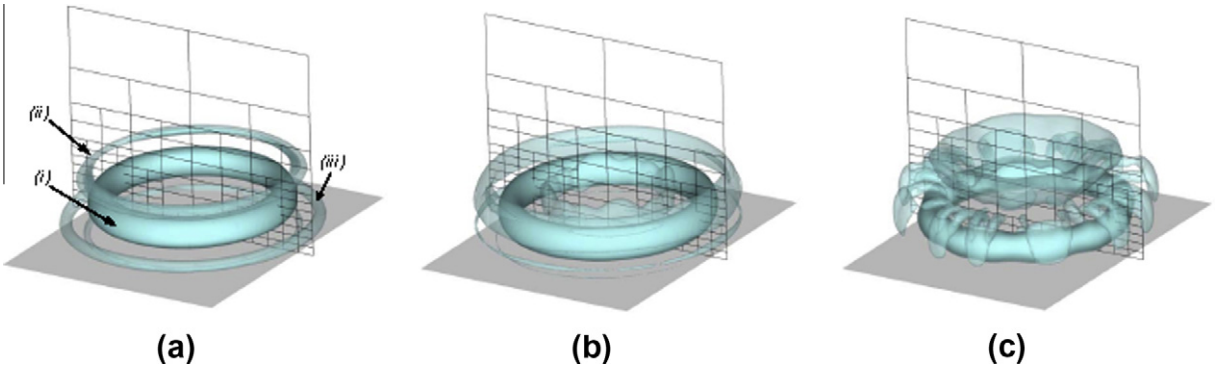

Fig. 13. Isosurfaces of Q for the case of a vortex impinging to a wall at three different times. The AMR block distribution is shown at an $y - z$ plane. (a) $t = 2.0$; (b) $t = 2.7$; (c) $t = 4.4$. (i), (ii) and (iii) indicate the primary, secondary and tertiary vortices, respectively.

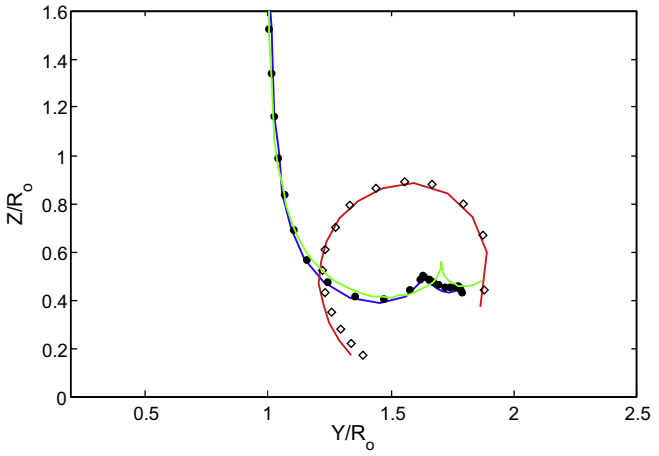

Fig. 14. Trajectories of the centers of the primary and secondary vortices at $Re_o = 570$. Symbols are from the uniform grid and lines from the AMR grid computation. • primary vortex; ◦ secondary vortex; green line is the trajectory of the primary vortex from [36] at $Re_o = 645$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

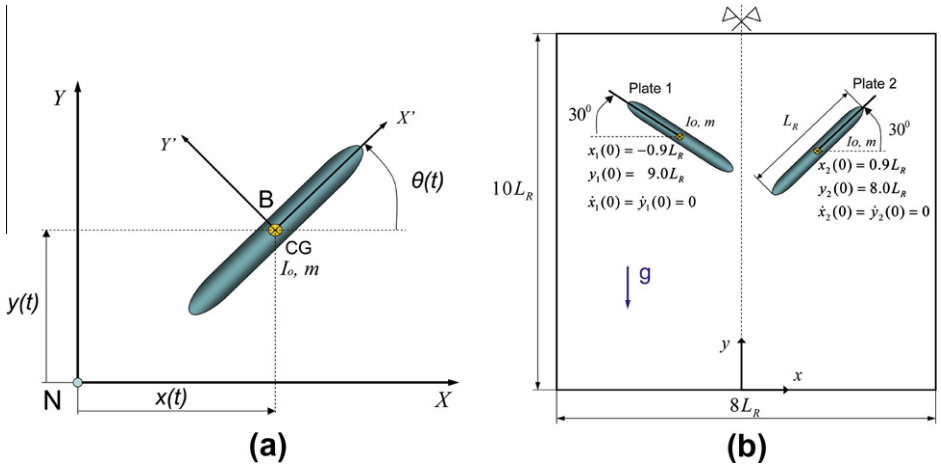

Fig. 15. (a) Variables describing two-dimensional rigid body motion for the falling plates. (b) Domain, boundary and initial conditions for the two falling plates problem.

$$\begin{bmatrix} [\mathbf{I}]_{(3\times3)} & & [\mathbf{0}]_{(3\times3)} & \\ & m & 0 & 0 \\ [\mathbf{0}]_{(3\times3)} & 0 & m & 0 \\ & 0 & 0 & I_o \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}}_{1\ (3\times1)} \\ \dot{\mathbf{q}}_{2\ (3\times1)} \end{Bmatrix} = \begin{Bmatrix} \mathbf{q}_{2\ (3\times1)} \\ f_x(\mathbf{q}_1, \mathbf{q}_2) \\ f_y(\mathbf{q}_1, \mathbf{q}_2) - mg \\ M_o(\mathbf{q}_1, \mathbf{q}_2) \end{Bmatrix}, \tag{22}$$

with $\mathbf{q}_1 = [x_{(t)} y_{(t)} \theta_{(t)}]^T$ and $\mathbf{q}_2 = \dot{\mathbf{q}}_1 \cdot x_{(t)}, y_{(t)}$ are the coordinates of the center of mass of the plate in the x and y directions, respectively, and $\theta_{(t)}$ its orientation angle (see Fig. 15a). f_x, f_y, are the corresponding hydrodynamics forces acting on the plate, and M_o, is their moment with respect to the center of mass. Eq. (22) has been made dimensionless using the chord length, c, the mean descent velocity of plate 2, U_R, and the fluid density, ρ_f, as reference variables. The thickness of each plate is 10% of the chord length, and their density is, $\rho_B = 5.1\rho_f$. Also $g = 3.0$, $m = 0.5$, and $I_o = 4.2 \times 10^{-2}$. The resulting Reynolds number is

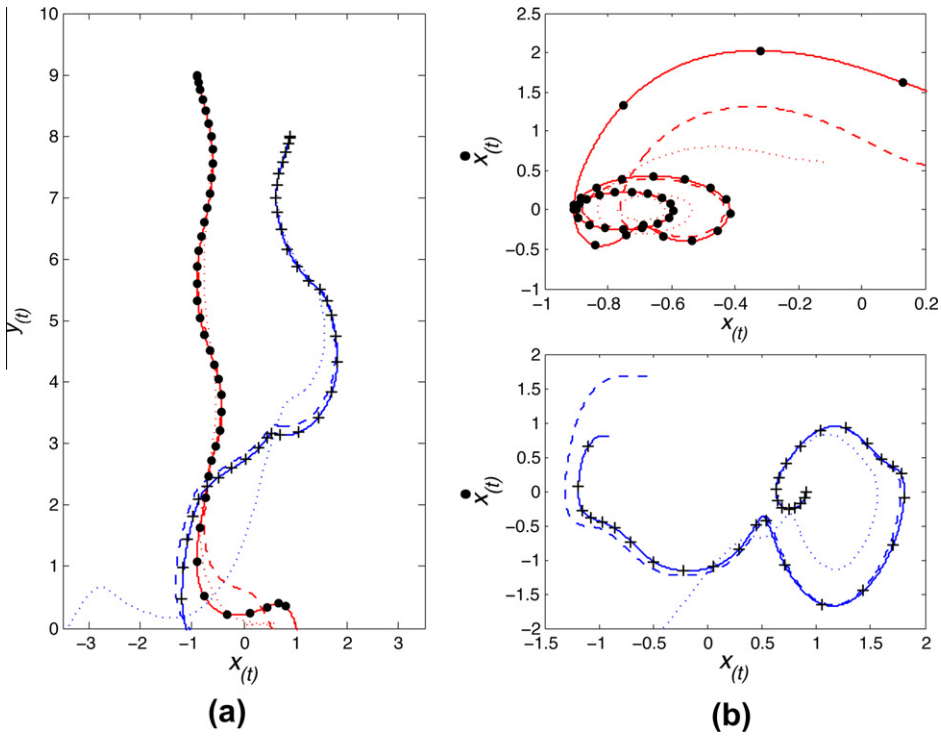

Fig. 16. (a) Position of center of mass for each plate as a function of time. (b) Phase diagrams for $x_{(t)}$. For the AMR computation: • plate 1; + plate 2. For the uniform grid computations: — 1024×1280; - 512×640; $\cdots$ 256×320, all lines are red for plate 1 and blue for plate 2. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

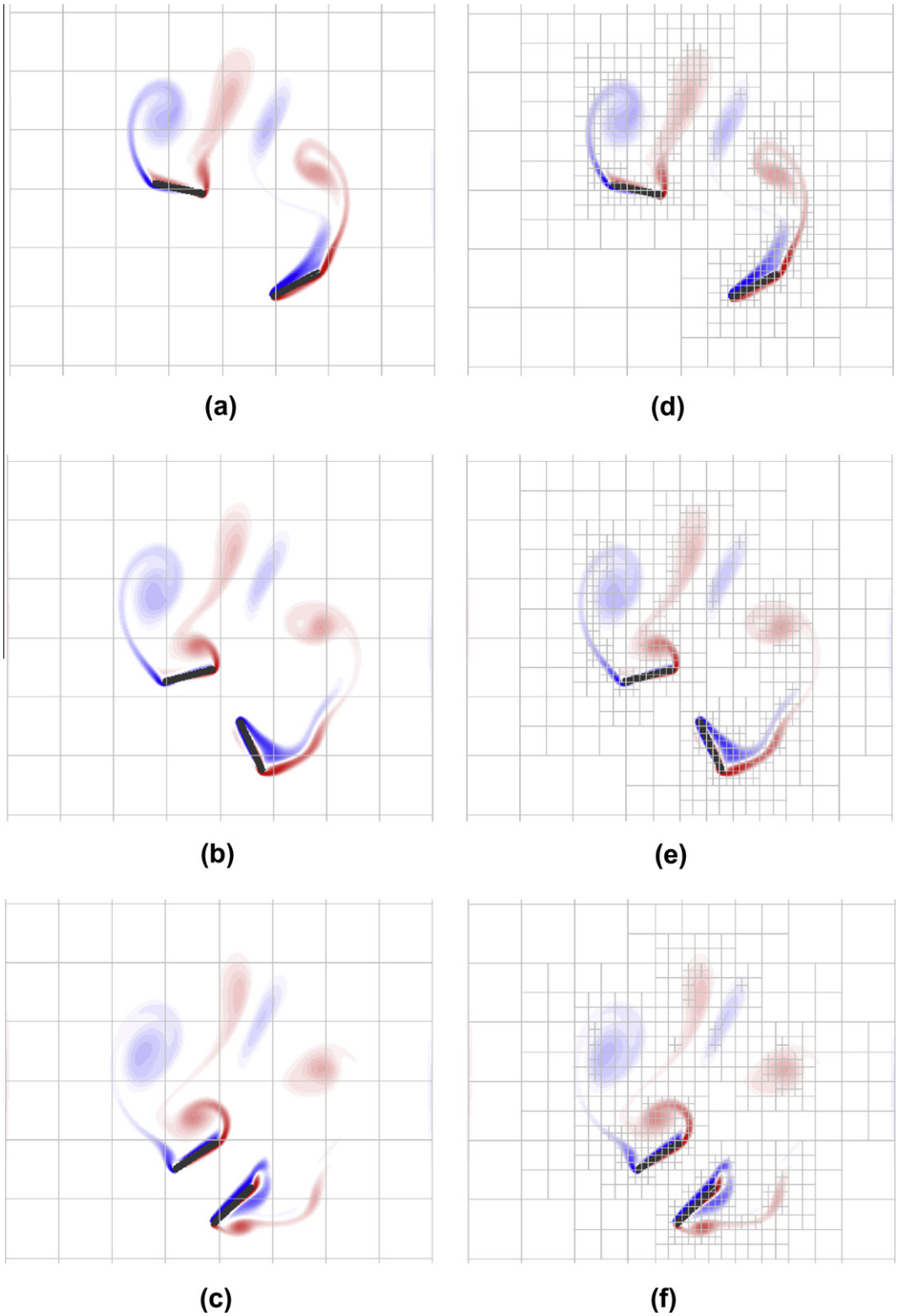

Fig. 17. Comparison between uniform (left side) and AMR (right side) grid computations for the case of the falling plates. Vorticity isolines are shown at (a) and (d) $t = 4.6$; (b) and (e) $t = 5.2$; (c) and (f) $t = 5.8$.

$Re = U_R c/\nu = 200$. The computational domain together with the initial conditions are shown in Fig. 15b. Both plates are initially at rest.

To evaluate the accuracy of the proposed AMR scheme two separate types of simulations are considered. First, three single level calculations using uniform grids of 1024×1280,512×640,256×320 nodes are conducted (the grid spacing on the finest grid is, $\Delta x = \Delta y = 0.0078 L_R$). Then, an AMR computation with four levels of refinement is carried out. Mesh adaptivity in this case was guided by two criteria: (i) the presence or not of a rigid body in a grid block; (ii) the vorticity modulus, $|\omega|$, as in the example given in the previous section. As a result the highest refinement level always surrounds the plates, as well as the areas of high velocity gradients in their wake. Note that grid cell size at the highest refinement level is the same as the one used in the finest uniform grid calculation. All computations were advanced in time for 22×10^3 steps, with $\Delta t = 2.0 \times 10^{-3}$.

The trajectories of the centers of mass for both bodies as a function of time, and phase diagrams of positions and velocities for the horizontal and vertical degrees-of-freedom are shown in Fig. 16a and b, respectively. Clearly grid resolution has a substantial effect on the trajectories of both plates, as manifested by the results on the three uniform grids. The AMR computation, on the other hand, where the grid is dynamically refined in areas of high velocity gradients, is in excellent agreement with the finest uniform grid, single-block computation.

A frame-by-frame comparison of the vorticity field in the last two computations is also shown in Fig. 17 with very good agreement as well. It is interesting to note the transition from steady fluttering fall to tumbling for plate 2 (see Fig. 17d and f). Tumbling motion has been observed by Andersen et al. [4] for a single falling plate under similar flow conditions, and usually occurs for $Re > 100$ and $I_o > 3 \times 10^{-2}$ [12]. For plate 1 transition from a fluttering fall to tumbling is limited by the wake of plate 2.

5.4. Flow around a Sphere at Re = 10,000

To test the performance of the AMR solver in turbulent flows, a LES of the flow around a sphere at $Re = U_\infty D/\nu = 10^4$ (U_∞ is the freestream velocity and D is the diameter of the sphere) is considered. This Reynolds number falls in the subcritical regime, where laminar boundary layer separation occurs on the surface of the sphere, and transition happens in the detached

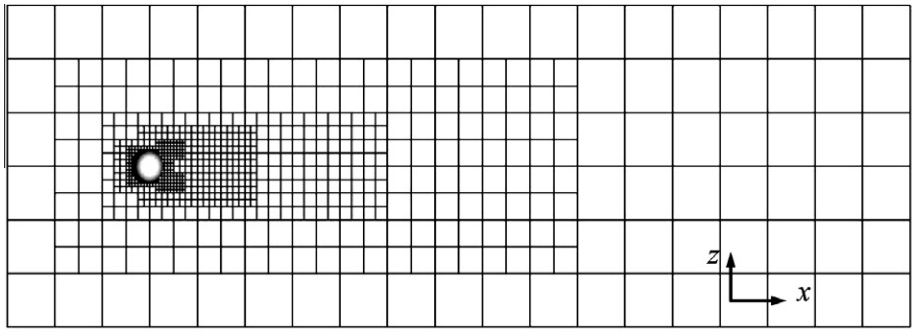

Fig. 18. AMR grid blocks for flow around a sphere at $Re = 10^4$. An $x - z$ plane at $y = 0$ is shown. Note that a window spanning from $-5 < x < 28$ in the x-direction and from $-5 < z < 5$ in the z-direction is shown. Six levels of refinement are used with 16^3 points per block.

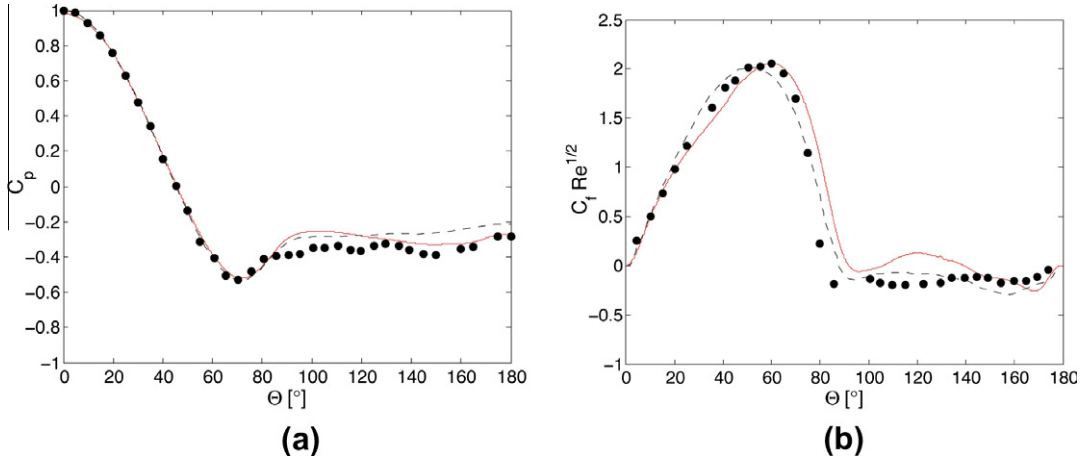

Fig. 19. Variation of (a) C_p, and (b) C_f on the surface of the sphere as a function of angle Θ from front stagnation point. Both are averaged in the azimuthal direction and time; — current AMR–LES calculation at $Re = 10^4$, - - - reference LES at $Re = 10^4$ [16], and • experiment at $Re = 1.6 \times 10^5$ [1].

shear layers in the wake. A rectangular domain is employed, spanning 38D in the streamwise direction and 24D in the cross stream directions. The sphere is centered at a distance of 10D downstream of the inflow boundary. The inflow velocity, U_∞, is prescribed at this boundary, and a convective condition used at the outflow boundary [32]. Slip wall boundary conditions are used in the other two directions.

A preliminary computation for laminar flow around a sphere at $Re = 300$ was done on a similar setting, using 5 levels of refinement and about 6.5×10^6 points. The resulting grid spacing close to the sphere was 0.0195D. The predicted mean drag coefficient is $C_D = 0.634$, and its oscillation amplitude $C_D' = 0.0039$. These values are in good agreement with the correspond-

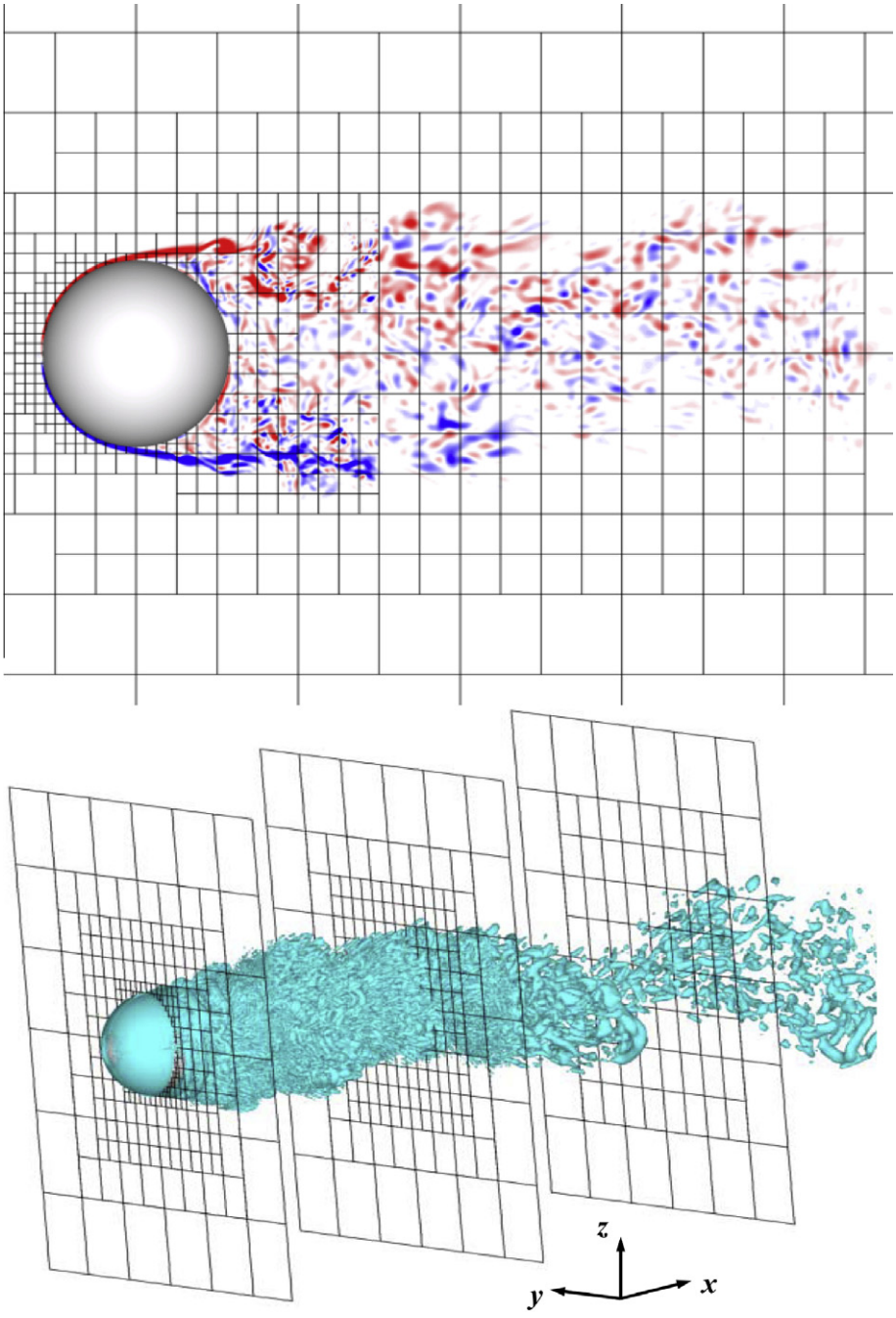

Fig. 20. A snapshot of the instantaneous flow field around the sphere at $Re = 10^4$. Top: Vorticity contours at an $x - z$ plane through the center of the sphere, $-20 < \overline{\omega}_y D/U_\infty < 20$, where (blue) and (red) indicate areas of counter-clockwise and clockwise rotation respectively. Bottom: Q-isosurfaces in the wake. The block distribution is also shown for 3 slices at positions $x = 0,3D,7D$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

ing results from computations by Johnson and Patel [22] of $C_D = 0.656$ and $C'_D = 0.0035$, respectively. The resulting Strouhal number is $St = 0.132$, which is also in good agreement with the value *0.137* reported in [22].

For the case at $Re = 10^4$ a finer grid with 6 levels of refinement is used with $11 \times 7 \times 7$ blocks at level 0. Each block consists of 16^3 cells. The grid is finest near the surface of the sphere and it is gradually derefined in the wake. The leaf-block arrangement at an $x - z$ plane passing through the center of the sphere is shown in Fig. 18. The cell sizes near the surface of the sphere are approximately, $\Delta x_i = 0.003D$, and the resulting distance of the first grid point off the wall is between $r^+ = 1.3 - 2.3$ (following [16], we define $r^+ = r\, u_\tau/\nu$, where the friction velocity is assumed to be $u_\tau = 0.04$). The number of points between the wall and $r^+ \sim 10$ is between 5 and 8. The total number of blocks is 18,400, and the number of leaf-blocks 16,184 corresponding to 66.3 million points. Further refinement was not attempted due to cost considerations. We should also note that temporal adaptivity was not utilized in this case since the location of the finest grid blocks needs to be in the laminar boundary layers on the sphere's surface which do not evolve with time.

The equations were integrated in time until the effect of initial conditions was eliminated. Statistics were accumulated over two shedding cycles, which is about $10U_\infty/D$ eddy turnover times or 20,000 timesteps. Although the sample is fairly small it is sufficient to provide reasonable estimates of the average force coefficients. LES of the flow around a sphere at the same Reynolds number performed by Constantinescu et al. [16], gave a mean drag coefficient $\overline{C}_D = 0.393$. The same value was reported in the LES by Yun et al. [45]. Our computation gave $\overline{C}_D = 0.405$, which is in very good agreement with the above results. In Fig. 19 the distribution of average pressure coefficient, $\overline{C}_p$, and skin friction coefficient, $\overline{C}_f$, are shown as a function of the azimuthal angle, Θ. The corresponding results from the LES by Constantinescu et al.[16] at the same Reynolds number, and the experiments by Achenbach [1] at a higher subcritical Reynolds number, $Re = 1.6 \times 10^5$, are also shown for comparison. In general the agreement is good, and $\overline{C}_p$ is practically the same in all three data sets for $0^\circ < \Theta < 90^\circ$. Both simulations also agree very well for $90^\circ < \Theta < 130^\circ$, and slightly over-predict $\overline{C}_p$ reported in the experiment. Further behind the sphere at, $130^\circ < \Theta < 180^\circ$, our $\overline{C}_p$ is lower by is approximately 6% of the peak value, probably due to our limited statistical sample.

The friction coefficient, $\overline{C}_f$, is also in good agreement with the reference data. In this case discrepancies among the different datasets are larger, due to larger errors in measuring this quantity as well as its sensitivity to numerical resolution. Our mean separation occurs at $\Theta = 91^\circ$ which is in very good agreement with the value of $\Theta = 90^\circ$ reported in [45]. Constantinescu et al. [16] is reported mean separation at $\Theta = 85^\circ$. They used spherical coordinates and a grid which is much finer near the surface of the sphere, while our resolution is comparable to the one utilized by Yun et al. [45].

In Fig. 20 an instantaneous snapshot of the spanwise vorticity, $\overline{\omega}_y$, is shown at an $x - z$ plane passing through the center of the sphere. The shear layer instability, roll-up and subsequent breakdown can be observed. The smooth variation of the vorticity between blocks at different refinement levels should also be noted, indicating that turbulent eddies are unaffected by the presence of block boundaries. A three-dimensional view of such eddies is also shown in Fig. 20, and is visualized using the second invariant of the velocity gradient tensor, Q. The AMR block structure at three different planes in the wake region ($x = 0$, $x = 3D$ and $x = 7D$) is also shown. As the refinement level is decreased for increasing x, the reduction in the range of scales resolved by the grid is evident.

6. Summary and conclusions

In the present study we propose a S-AMR formulation for the simulation of fluid–structure interaction problems in viscous incompressible flows. A single-block solver is employed on a hierarchy of sub-grids with varying spatial resolution. Each of these sub-grid blocks has a structured Cartesian topology, and is part of a tree data-structure that covers the entire computational domain. Time advancement is done using a fractional-step method. All spatial derivatives are approximated with second-order finite-differences on a staggered grid. The Paramesh toolkit [25] is utilized to keep track of the grid hierarchy, and perform the required restriction/prolongation and guard-cell filling operations. Coupling the proposed AMR algorithm with the embedded-boundary approach proposed in [41] results in a robust and cost/efficient tool applicable to complex fluid structure interaction problems.

We demonstrated that the accuracy of dynamic AMR is greatly effected by the conservation properties of the prolongation and restriction operators. We developed a divergence-preserving prolongation operator tailored to the specific AMR topology, where the grid size between consecutive refinement levels can only differ by a factor of two. Overall the second-order spatial and temporal accuracy of the basic solver are maintained. We have also shown that our scheme is well suited for eddy resolving computations such as LES and DNS. In the former case the derefinement jumps normal to the main advection direction may lead to high interpolation and aliasing errors due to the fact that a large portion of small structures being advected can not be resolved by the coarse grid. We found that the strategy proposed in [40], where the LDEV model is used to parameterize the SGS stresses with a varying filter size at refinement interfaces, together with explicit filtering of the advective term works very well with the present AMR scheme, and gave excellent results for the case of the flow around a sphere at $Re = 10^4$.

Compared to available structured [27] or unstructured AMR [21] formulations, the constraint that neighboring blocks can only differ by one level of refinement may result in larger overall grids to achieve the same local resolution especially in internal flows. This is not a major concern in LES of turbulent and transitional flows, where grid discontinuities generated by neighboring blocks that differ by more than a factor of two can contaminate the high frequency content and unphysically enhance turbulent fluctuations on the fine-grid side, and are, therefore, not desirable.

The computational efficiency of the proposed formulation is a balancing act between the lower CPU and memory cost incurred using AMR, and the additional work derived from the augmented computational complexity. In general for high Reynolds number FSI problems, where the fine-grid patches needed to resolve the boundary layers on the surface of the body have to be constantly rearranged following the body's motion, the proposed solver has a significant advantage over single-block solvers. For the falling plates simulation, for example, the number of grid points utilized in the AMR computation was 1/10 of the ones used in the uniform grid solution. In terms of wall time, the AMR calculation took about 3 s per timestep on a single processor, while the equivalent uniform grid run required roughly twice that amount. We should also note that the uniform grid solver utilizes a direct solver for the Poisson equation (see [5] for details), which is much more efficient compared to the multigrid solver. If a multigrid solver was to be adopted for the case of the uniform grid too, the computational savings using AMR would have been an order of magnitude higher. We are currently conducting detailed test on the parallel performance of the AMR solver and we will report these results in a future article.

Acknowledgments

This research was supported by the National Science Foundation (Grant Nos. OCI-0904920 and CBET-0932613).

Appendix A. Divergence-preserving prolongation in three-dimensions

The development of divergence-preserving operators in three-dimensions is similar to the one for two-dimensions outlined in Section 3.2. In particular, the discrete divergence of the coarse-grid cell shown in Fig. 21 is

$$\frac{u^l_{i,j,k}-u^l_{i-1,j,k}}{\Delta x^l}+\frac{v^l_{i,j,k}-v^l_{i,j-1,k}}{\Delta y^l}+\frac{w^l_{i,j,k}-w^l_{i,j,k-1}}{\Delta z^l}=D\mathbf{u}. \tag{23}$$

To interpolate the velocities on the cell faces, the one-dimensional interpolations in two-dimensions, are now replaced with two-dimensional quadratic interpolations assuming a polynomial variation of the form:

$$u(\xi,\eta)=a_0+a_1\xi+a_2\eta+a_3\xi\eta+a_4\xi^2+a_5\eta^2+a_6\xi^2\eta+a_7\xi\eta^2+a_8\xi^2\eta^2. \tag{24}$$

An example stencil is shown in Fig. 21a, and a system analogous to (7) can be constructed by applying Eq. (24) in the ($\xi_{i+\alpha}$, $\eta_{j+\beta}$), α, $\beta=-1,0,1$ positions at level l, except for (ξ_i, η_j). Then the known value of the coarse-grid variable at location (ξ_i, η_j) is used in

$$u^l_{i,j,k}=\frac{1}{4}\left(u^{l+1}_{i',j',k'}+u^{l+1}_{i',j'+1,k'}+u^{l+1}_{i',j'+1,k'+1}+u^{l+1}_{i',j',k'+1}\right), \tag{25}$$

and the four fine-grid values are computed by interpolations from Eq. (24). The matrix associated to the resulting system can be inverted using symbolic manipulation software, such that the computation of the interpolation coefficients $a_i, i=0,1,\ldots,8$ will require a 9×9 matrix–vector multiplication. The computation of the four fine-grid variables requires four additional 9×1 vector–vector multiplications.

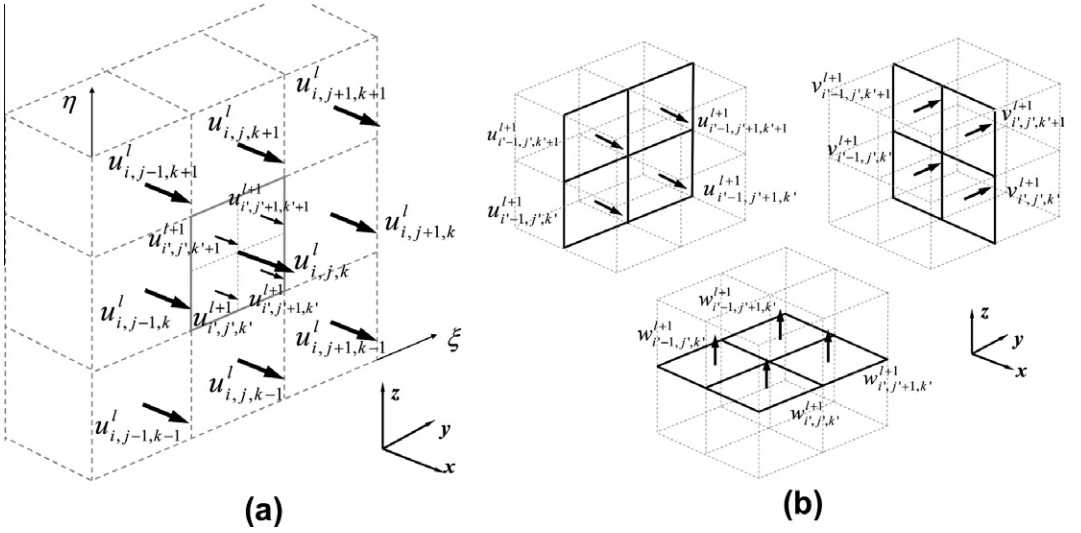

Fig. 21. (a) Interpolation stencil for 2D face interpolations used in three-dimensional divergence-preserving prolongation. (b) Internal variables to a given coarse-grid cell indexed by *i*, *j*, *k*.

Once the velocity field has been found on the cell faces, the velocities internal to the coarse cell need to be obtained. There are twelve internal fine-grid variables, $u^{l+1}_{i'-1,j',k'}, u^{l+1}_{i'-1,j'+1,k'}, u^{l+1}_{i'-1,j',k'+1}, u^{l+1}_{i'-1,j'+1,k'+1}, v^{l+1}_{i',j',k'}, v^{l+1}_{i'-1,j',k'}, v^{l+1}_{i',j',k'+1}, v^{l+1}_{i'-1,j',k'+1}, w^{l+1}_{i',j',k'}, w^{l+1}_{i'-1,j',k'}$, $w^{l+1}_{i'-1,j'+1,k'}$ and $w^{l+1}_{i',j'+1,k'}$, that need to be determined (see Fig. 21b). Four of the twelve $\left(u^{l+1}_{i'-1,j',k'}, u^{l+1}_{i'-1,j'+1,k'+1}, v^{l+1}_{i'-1,j',k'}, v^{l+1}_{i',j',k'+1}\right)$, can be found using one-dimensional quadratic interpolation from the corresponding coarse-face variables, and for the remaining ones, eight discrete divergence equations are used as in the two-dimensional example. The resulting system can be written as

$$\begin{bmatrix} 0 & 0 & 0 & 0 & \gamma & 0 & 0 & 0 \\ 0 & 0 & \beta & 0 & 0 & \gamma & 0 & 0 \\ 0 & \alpha & 0 & 0 & 0 & 0 & \gamma & 0 \\ 0 & -\alpha & -\beta & 0 & 0 & 0 & 0 & \gamma \\ \alpha & 0 & 0 & \beta & -\gamma & 0 & 0 & 0 \\ -\alpha & 0 & 0 & 0 & 0 & -\gamma & 0 & 0 \\ 0 & \alpha & 0 & -\beta & 0 & 0 & -\gamma & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\gamma \end{bmatrix} \begin{Bmatrix} u^{l+1}_{i'-1,j',k'+1} \\ u^{l+1}_{i'-1,j'+1,k'} \\ v^{l+1}_{i',j',k'} \\ v^{l+1}_{i'-1,j',k'+1} \\ w^{l+1}_{i'-1,j',k'} \\ w^{l+1}_{i',j',k'} \\ w^{l+1}_{i'-1,j'+1,k'} \\ w^{l+1}_{i',j'+1,k'} \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \end{Bmatrix}, \tag{26}$$

where $\alpha = 1/\Delta x^{l+1}$,$\beta = 1/\Delta y^{l+1}$ and $\gamma = 1/\Delta z^{l+1}$. The coefficients in the right hand side are given by

$$\begin{aligned}
b_1 &= D\mathbf{u} - \frac{u^{l+1}_{i'-1,j',k'} - u^{l+1}_{i'-2,j',k'}}{\Delta x^{l+1}} - \frac{v^{l+1}_{i'-1,j',k'} - v^{l+1}_{i'-1,j'-1,k'}}{\Delta y^{l+1}} + \frac{w^{l+1}_{i'-1,j',k'-1}}{\Delta z^{l+1}}, \\
b_2 &= D\mathbf{u} - \frac{u^{l+1}_{i',j',k'} - u^{l+1}_{i'-1,j',k'}}{\Delta x^{l+1}} + \frac{v^{l+1}_{i',j'-1,k'}}{\Delta y^{l+1}} + \frac{w^{l+1}_{i',j',k'-1}}{\Delta z^{l+1}}, \\
b_3 &= D\mathbf{u} + \frac{u^{l+1}_{i'-2,j'+1,k'}}{\Delta x^{l+1}} - \frac{v^{l+1}_{i'-1,j'+1,k'} - v^{l+1}_{i'-1,j',k'}}{\Delta y^{l+1}} + \frac{w^{l+1}_{i'-1,j'+1,k'-1}}{\Delta z^{l+1}}, \\
b_4 &= D\mathbf{u} - \frac{u^{l+1}_{i',j'+1,k'}}{\Delta x^{l+1}} - \frac{v^{l+1}_{i',j'+1,k'}}{\Delta y^{l+1}} + \frac{w^{l+1}_{i',j'+1,k'-1}}{\Delta z^{l+1}}, \\
b_5 &= D\mathbf{u} + \frac{u^{l+1}_{i'-2,j',k'+1}}{\Delta x^{l+1}} + \frac{v^{l+1}_{i'-1,j'-1,k'+1}}{\Delta y^{l+1}} - \frac{w^{l+1}_{i'-1,j',k'+1}}{\Delta z^{l+1}}, \\
b_6 &= D\mathbf{u} - \frac{u^{l+1}_{i',j',k'+1}}{\Delta x^{l+1}} - \frac{v^{l+1}_{i',j',k'+1} - v^{l+1}_{i',j'-1,k'+1}}{\Delta y^{l+1}} - \frac{w^{l+1}_{i',j',k'+1}}{\Delta z^{l+1}}, \\
b_7 &= D\mathbf{u} + \frac{u^{l+1}_{i'-2,j'+1,k'}}{\Delta x^{l+1}} - \frac{v^{l+1}_{i'-1,j'+1,k'+1}}{\Delta y^{l+1}} - \frac{w^{l+1}_{i'-1,j'+1,k'+1}}{\Delta z^{l+1}}, \\
b_8 &= D\mathbf{u} - \frac{u^{l+1}_{i',j'+1,k'+1} - u^{l+1}_{i'-1,j'+1,k'+1}}{\Delta x^{l+1}} - \frac{v^{l+1}_{i',j'+1,k'+1} - v^{l+1}_{i',j',k'+1}}{\Delta y^{l+1}} - \frac{w^{l+1}_{i',j'+1,k'+1}}{\Delta z^{l+1}}.
\end{aligned} \tag{27}$$

As this system is of rank 7, one more equation similar to (9) is added. As in the two-dimensional case, the left-pseudo-inverse of the resulting 9×8 coefficient matrix A_e can be computed symbolically. The implementation requires additional 8×9 matrix–vector operations to obtain the unknown variables.

References

[1] E. Anchenbach, Experiments on flow past spheres at very high Reynolds-numbers, J. Fluid Mech. 54 (565) (1972).
[2] A.S. Almgren, J.B. Bell, P. Colella, L.H. Howell, M.L. Welcome, A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations, J. Comput. Phys. 142 (1) (1998) 1–46.
[3] A.S. Almgren, J.B. Bell, W.Y. Crutchfield, Approximate projection methods: part I. Inviscid analysis, SIAM J. Sci. Comput. 22 (4) (2000) 1139–1159.
[4] A. Andersen, U. Pesavento, Z.J. Wang, Unsteady aerodynamics of fluttering and tumbling plates, J. Fluid Mech. 541 (2005) 65–90.
[5] E. Balaras, Modeling complex boundaries using an external force field on fixed cartesian grids in large-eddy simulations, Comput Fluids 33 (2004) 375–404.
[6] E. Balaras, U. Piomelli, J.M. Wallace, Self-similar states in turbulent mixing layers, J. Fluid Mech. 446 (2001) 1–24.
[7] D.S. Balsara, Divergence free adaptive mesh refinement for MHD, J. Comput. Physics 174 (2) (2001) 614–648.
[8] Haim Baruh, Analytical Dynamics, McGraw-Hill Science, 1999. p. 718.
[9] S. Bayyuk, K. Powellt, B. van Leer, A Simulation Technique for 2-d Unsteady Inviscid Flows Around Arbitrarily Moving and Deforming Bodies of Arbitrary Geometry, AIAA-93-3391-CP, 1993, pp. 1013–1024.
[10] J. Bell, M. Berger, J. Saltzman, M. Welcome, Three-dimensional adaptive mesh refinement for hyperbolic conservation laws, SIAM J. Sci. Comput. 15 (1) (1994) 127–138.
[11] J. Bell, P. Colella, H. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, J. Comput. Phys. 85 (1989) 257–283.
[12] A. Belmonte, H. Eisenberg, E. Moses, From flutter to tumble: inertial drag and froude similarity in falling paper, Phys. Rev. Lett. 81 (2) (1998) 345–348.
[13] M. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1989) 64–84.

[14] M. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comput. Phys. 53 (1984) 484–512.
[15] Jung-Il. Choi, Roshan C. Oberoi, Jack R. Edwards, Jacky A. Rosati, An immersed boundary method for complex incompressible flows, J. Comput. Phys. 224 (2007) 757–784.
[16] G. Constantinescu, M. Chapelet, K. Squires, Turbulence modeling applied to flow over a sphere, AIAA J. 41 (9) (2003) 1733–1742.
[17] T.L. Doligalski, C.R. Smith, J.D.A. Walker, Vortex interactions with walls, Annu. Rev. Fluid Mech. 26 (1994) 573–616.
[18] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (1) (2000) 35–60.
[19] B. Fryxell, K. Olson, P. Ricker, F.X. Timmes, M. Zingale, D.Q. Lamb, P. MacNeice, R. Rosner, J.W. Truran, H. Tufo, Flash: an adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes, Astrophys. J. Suppl. 131 (2000) 273–334.
[20] Boyce E. Griffith, Richard D. Hornung, David M. McQueen, Charles S. Peskin, An adaptive, formally second order accurate version of the immersed boundary method, J. Comput. Phys. 223 (1) (2007) 10–49.
[21] F. Ham, F. Lien, A. Strong, A cartesian grid method with transient anisotropic adaptation, J. Comput. Phys. 179 (2002) 469–494.
[22] T.A. Johnson, V.C. Patel, Flow past a sphere up to a Reynolds number of 300, J. Fluid Mech. 378 (1999) 19–70.
[23] J. Van Kan, A second-order accurate pressure-correction scheme for viscous incompressible flow, SIAM J. Sci. Stat. Comput. 7 (3) (1986) 870–891.
[24] Dongwook Lee, Anil E Deane, An unsplit staggered mesh scheme for multidimensional magnetohydrodynamics, J. Comput. Phys. 228 (4) (2009) 952–975.
[25] P. MacNeice, K.M. Olson, C. Mobarry, R. deFainchtein, C. Packer, Paramesh: a parallel adaptive mesh refinement community toolkit, Comput. Phys. Commun. 126 (2000) 330–354.
[26] D. Martin, K. Cartwright, Solving Poisson's Equation Using Adaptive Mesh Refinement, LBL Research Report, 1996.
[27] Daniel F. Martin, Phillip Colella, Daniel Graves, A cell-centered adaptive projection method for the incompressible Navier–Stokes equations in three dimensions, J. Comput. Phys. 227 (3) (2008) 1863–1886.
[28] D. Mavriplis, Unstructured Mesh Generation and Adaptivity, ICASE Report No. 95-26, 1995.
[29] C. Meneveau, T.S. Lund, W.H. Cabot, A lagrangian dynamic subgrid-scale model of turbulence, J. Fluid Mech. 319 (1996) 353–385.
[30] R. Mittal, G. Iaccarino, Immersed boundary methods, Annu. Rev. Fluid Mech. 37 (2005) 239–261.
[31] P. Orlandi, R. Verzicco, Vortex rings impinging on walls – axisymmetrical and 3-dimensional simulations, J. Fluid Mech. 256 (1993) 615–646.
[32] I. Orlanski, A simple boundary condition for unbounded hyperbolic flows, J. Comput. Phys. 21 (3) (1976) 251–269.
[33] C.S. Peskin, Flow patterns around heart valves – numerical method, J. Comput. Phys. 10 (2) (1972) 252.
[34] 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.
[35] F. Sarghini, U. Piomelli, E. Balaras, Scale-similar models for large-eddy simulations, Phys. Fluids 11 (1999) 1596–1607.
[36] J.D. Swearingen, J.D. Crouch, R.A. Handler, Dynamics and stability of a vortex ring impacting a solid boundary, J. Fluid Mech. 297 (1995) 1–28.
[37] T. Tezduyar, Finite element methods for flow problems with moving boundaries and interfaces, Arch. Comput. Meth. Eng. 8 (2) (2001) 83–130.
[38] H.S. Udaykumar, W. Shyy, M.M. Rao, Elafint: a mixed eulerian-lagrangian method for fluid flows with complex and moving boundaries, Int. J. Numer. Meth. Fluids 22 (8) (1996) 691–712.
[39] M. Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, J. Comput. Phys. 209 (2) (2005) 448–476.
[40] M. Vanella, U. Piomelli, E. Balaras, Effect of grid discontinuities on large-eddy simulation statistics and flow fields, J. Turbul. 9 (32) (2008) 1–23.
[41] M. Vanella, E. Balaras, A moving-least-squares reconstruction for embedded-boundary formulations, J. Comput. Phys. 228 (18) (2009) 6617–6628.
[42] J.D.A. Walker, C.R. Smith, A.W. Cerra, T.L. Doligalski, The impact of a vortex ring on a wall, J. Fluid Mech. 181 (1987) 99–140.
[43] J. Yang, S. Preidikman, E. Balaras, A strongly coupled, embedded-boundary method for fluid–structure interactions of elastically mounted rigid bodies, J. Fluid Struct. 24 (2) (2008) 167–182.
[44] J.M. Yang, E. Balaras, An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries, J. Comput. Phys. 215 (2006) 12–40.
[45] G. Yun, D. Kim, H. Choi, Vortical structures behind a sphere at subcritical Reynolds numbers, Phys. Fluids 18 (1) (2006) 015102.