Finite-difference lattice Boltzmann method with a block-structured adaptive-mesh-refinement technique

Abbas Fakhari[*] and Taehun Lee[†]

Department of Mechanical Engineering, The City College of the City University of New York, New York 10031, USA

(Received 6 August 2013; revised manuscript received 23 December 2013; published 19 March 2014)

An adaptive-mesh-refinement (AMR) algorithm for the finite-difference lattice Boltzmann method (FDLBM) is presented in this study. The idea behind the proposed AMR is to remove the need for a tree-type data structure. Instead, pointer attributes are used to determine the neighbors of a certain block via appropriate adjustment of its children identifications. As a result, the memory and time required for tree traversal are completely eliminated, leaving us with an efficient algorithm that is easier to implement and use on parallel machines. To allow different mesh sizes at separate parts of the computational domain, the Eulerian formulation of the streaming process is invoked. As a result, there is no need for rescaling the distribution functions or using a temporal interpolation at the fine-coarse grid boundaries. The accuracy and efficiency of the proposed FDLBM AMR are extensively assessed by investigating a variety of vorticity-dominated flow fields, including Taylor-Green vortex flow, lid-driven cavity flow, thin shear layer flow, and the flow past a square cylinder.

DOI: 10.1103/PhysRevE.89.033310 PACS number(s): 47.11.Qr

I. INTRODUCTION

The adaptive-mesh-refinement (AMR) technique is a logical extension of the concept behind invoking nonuniform grids to efficiently utilize computer resources. Grids of higher resolution are used wherever a more accurate solution is desired. There are numerous scientific and engineering applications with large variations in the scale of their problems of interest, for which AMR routines are most suited. For instance, in complex fluids, higher resolution is preferred near the interface between different phases because of the rapid changes in the fluid properties.

The concept of AMR for structured grids was pioneered by Berger and co-workers [1,2]. Berger and Colella [2] proposed a grid-based AMR for solving two-dimensional hyperbolic equations in a conservative form. Rectangular, overlapping grid patches were placed in regions with a need for higher resolution. Each patch consisted of a varying number of cells of the same size. This strategy, however, is not flexible in adaptivity, nor is it optimal in terms of computational cost and memory management because the overlapping rectangular patches may cover parts of the computational domain that do not need to be resolved, resulting in a waste of computer memory and time. Moreover, the entire grid hierarchy must be rebuilt once the refinement-derefinement is carried out. de Zeeuw and Powel [3] proposed an AMR scheme with a quad-tree data structure in two dimensions (2D). The idea was to bisect each cell, once it was flagged for refinement, along the coordinate directions. They claimed that finding the neighbors of a given cell was superior to storing eight integer words per cell. Consequently, they preferred to perform a tree search to find the neighboring cells rather than storing the cell-neighbor information via pointers, which cost them approximately 25% of their computation time. To overcome the computational overhead of the tree-based approach, a fully threaded tree (FTT) AMR was proposed by Khokhlov [4]. In the FTT algorithm [4,5], the computational domain is covered by cubic cells (octs) in 3D. Each oct has eight child cells arranged in an oct-tree data structure. Although tree traversal to find the nearest neighbors of a given cell is avoided using pointers and linked lists, the tree structure must still be maintained and modified for traversing the threaded tree during search operations. The FTT data structure is, however, simpler and more efficient than the tree-based approach. Compared to an equivalent uniform mesh, a minimum factor of 10 savings in memory and computation time was reported using FTT [4]. Recently, Ji *et al.* [6] proposed a cell-based structured AMR that replaces the tree structure with a hash table. Accordingly, the computer memory required for storing the data structure is reduced. The penalty paid for this savings in memory is of course the computation overhead arising from the calculation of the connectivity information, such as finding the neighbors of a particular cell. Needless to say, the hashing technique relies on external resources for maintaining the data structure and connectivity information, which prevents the user from having a transparent and stand-alone AMR routine.

There are numerous AMR packages available [7–12] in the literature. Among them, PARAMESH [7] is designed to extend an existing serial code with a uniform mesh to a parallel code with AMR capabilities. It uses a hierarchy of structured blocks containing a fixed number of cells. While the data structure is that of a tree, the constituent elements are regular blocks that contain a fixed number of computational cells as opposed to an irregular number of cells in the grid-based AMR [2]. Instead of storing the connectivity information for a single cell, an array of cells within each block shares the data, reducing the memory overhead. On the downside, however, time-consuming tree traversal and search operations must still be performed to find the neighbors of a given block. Constructing a parallel AMR algorithm is yet another demanding issue. There is ongoing research concerning paralleling AMR algorithms and the load balancing techniques associated with them. For further information on the AMR and its challenges, the interested reader is referred to an excellent discourse by Rantakokko and Thuné [13].

[*]fakhari81@gmail.com
[†]thlee@ccny.cuny.edu; http://www.ccny.cuny.edu/profiles/Taehun-Lee.cfm

Over the past two decades, the lattice Boltzmann method (LBM) has become increasingly popular [14]. It has been improved and utilized by many researchers in different areas [15]. Different strategies have been used to facilitate implementation of the LBM on nonuniform grids. In addition to interpolation-supplemented LBM [16,17], a variety of finite-difference [18–20], finite-volume [21–23], and finite-element [24] lattice Boltzmann equation (LBE) models have been proposed. Most of the available LBE methods deal with local grid refinement with static refinement of the mesh at the beginning of simulation [25–28]. Few studies, however, have been focused on developing a dynamically refined grid for lattice Boltzmann models. Tölke *et al.* [29] proposed such a scheme for the simulation of multiphase flows. They invoked a data structure containing a hierarchy of tree-type grids. Yu and Fan [30] used the PARAMESH toolkit [7] to develop an AMR LBM for the simulation of a bubble rising under buoyancy. Recently, a stencil adaptive LBM in 2D was proposed by Wu and Shu [31]. The idea is to combine two types of symmetric stencils to achieve adaptivity for a nine-velocity lattice in 2D. Chen *et al.* [32] proposed an interpolation-supplemented LBM on quad-tree grids. They used a back-and-forth error compensation and correction algorithm to maintain the second-order accuracy of the streaming step while invoking a linear, rather than quadratic, interpolation. Most recently, Eitel-Amor *et al.* [33] developed an AMR LBM with a cell-centered data structure, as opposed to the conventional LBM with vertex points.

Most of the above-mentioned AMR LBM schemes use different time steps on grids with different lattice sizes. This requires storing additional data from the previous time step(s) to perform temporal interpolation for coarse grids. The aforementioned methods modify the relaxation time, according to the grid spacing, to keep the viscosity constant in the fine and coarse grids. This approach has two evident shortcomings. First, the maximum grid ratio, which is the ratio of the coarsest and finest grid spacings, is limited by the choice of the relaxation time. Increasingly larger relaxation times are needed on the finer grids and smaller relaxation rates are required on the coarser grids and these choices are restricted by the constraints of a small Knudsen number and numerical stability, respectively, which in turn limit the maximum allowable grid aspect ratio. Second, to maintain a continuous solution across grid boundaries, rescaling the distribution functions on the borders of fine-coarse grids is inevitable.

In the present study, we propose a lattice Boltzmann model with an adaptive-mesh-refinement strategy. The aim of this study is twofold. On the one hand, an efficient and straightforward algorithm for AMR is proposed that does not need a tree structure to maintain the connectivity data between the blocks. The proposed AMR algorithm alone can be used to solve partial differential equations including, but not limited to, the Navier-Stokes equations. On the other hand, in conjunction with the AMR, a finite-difference lattice Boltzmann method (FDLBM) is developed. Unlike the previous multiblock LBMs, the current model does not manipulate the speed of sound (as is the case in, e.g., [26]), nor does it need to rescale the distribution functions at the borders of fine-coarse blocks. All of the blocks at different refinement levels are advanced in time with the same time step but different Courant-Friedrichs-Lewy (CFL) numbers depending on the grid resolution. Consequently, temporal interpolation of the solution on the coarser blocks is not required to provide the boundary data for the finer blocks. It is worth mentioning that comparison of the proposed FDLBM with the standard LBM on nonuniform grids is not covered in this study.

The rest of the paper is organized as follows. The AMR algorithm will be elaborated in Sec. II, followed by a description of the FDLBM in Sec. III. The verification and validation of the proposed FDLBM AMR is presented in Sec. IV. This includes the evaluation of the convergence rate for the Taylor-Green decaying vortex flow, comparison of error indicators in the simulations of cavity flow and Kelvin-Helmholtz instability of a shear layer, and transient flow over a square cylinder. The paper is concluded in Sec. V with a brief summary.

II. ADAPTIVE-MESH REFINEMENT

We start by describing the architecture of our AMR scheme. For the sake of simplicity in illustration and explanation, we elaborate the AMR algorithm in 2D. The corner stone of the present AMR is a rectangular block with $n_x \times n_y$ vertex nodes, as depicted in Fig. 1. Initially, the computational domain is covered by a single block or, depending on the geometry, multiple blocks, referred to as the root block. The root block is obviously too coarse to start with; therefore, it is flagged for refinement. Upon refinement, four child blocks (eight child blocks in 3D) are created. The refinement process is continued until the maximum refinement level $l_{\max}$ is reached. The blocks at the highest refinement level are called leaf blocks because they have no children. The constructing blocks are self-similar in the sense that they have the same number of vertex points but different grid spacings depending on their hierarchical levels. The root block has a refinement level of $l = 0$, while the refinement level associated with the finest block is $l = l_{\max}$,

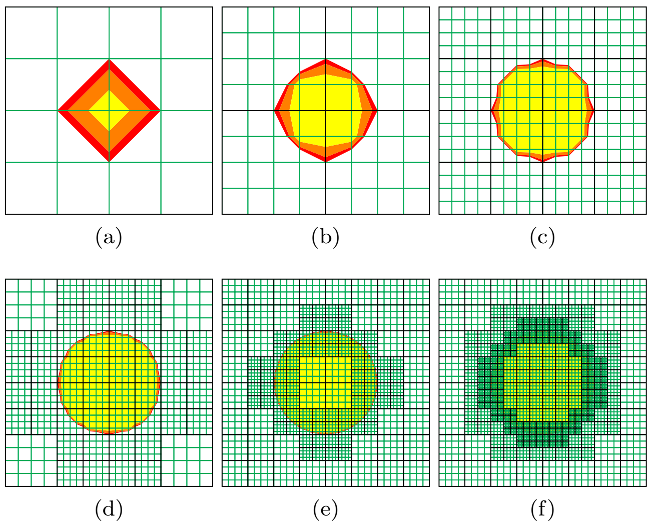


FIG. 1. (Color online) Block-structured AMR: configuration of the blocks (black lines) containing an interface (red-color contour) for (a) $l_{\max} = 0$, the root block with $(n_x \times n_y) = 5 \times 5$ vertex nodes (green lines); (b) $l_{\max} = 1$; (c) $l_{\max} = 2$; (d) $l_{\max} = 3$; (e) $l_{\max} = 4$; and (f) $l_{\max} = 5$, final arrangement of the blocks.

where $l_{\max}$ is an input parameter usually chosen in the range $l_{\max} \sim 5$–8. The grid spacings can be computed from the size of the root block in the x and y directions, L_x and L_y, by

$$\Delta x = \frac{L_x}{(n_x - 1)2^l}, \quad \Delta y = \frac{L_y}{(n_y - 1)2^l}. \tag{1}$$

It is clear that the resolution increases by a factor of 2 when moving from a parent block to its child at the next refinement level. It is worth mentioning that, in the LBM, all the quantities in Eq. (1) are integers; therefore, the grid spacings can be determined at no cost whenever necessary. However, it is preferable to store the connectivity information. Each block, which is a derived type in FORTRAN, stores its parent identification (ID) and its level of refinement. It also holds four pointers to its children, eight pointers to its orthogonal and diagonal neighbors, and a logical variable that indicates whether it is a leaf block. Unlike the FTT [5] or the cell-based AMR [6], we keep track of all the pointers to neighbors and of all the children IDs of a given block. This increases the performance of the code by avoiding "if statements" or "lookup tables" to find the neighbors of a certain block because the position of a block and its refinement level are required to find the neighboring blocks. We will further explain the key differences later on.

A block of cells has some advantages over a single cell. First, the memory required to store connectivity information and other necessary data is significantly reduced because each block has $(n_x - 1) \times (n_y - 1)$ cells that share the same attributes. Second, there is no need to refine additional cell layers around a refined cell to maintain a smooth solution across a fine-coarse boundary; this is automatically taken care of using the block-structured scheme. To have a smooth solution, however, a constraint requiring at most one level jump between the refinement levels of neighboring blocks is imposed in both orthogonal and diagonal directions.

The AMR hierarchy is built by noting that the neighbor-neighbor relation between two adjacent blocks with different levels is not reciprocal [4]. For example, in Fig. 2(a), the east neighbor of block A2 is block B, while the west neighbor of block B is not block A2. Indeed, the west neighbor of block B is block A (A2's parent block), which is at the same refinement level as block B. That is, the neighbors of a given block can be at the same level or at a lower level, but not at a higher refinement level; otherwise some blocks will end up having two neighbors on at least one side, which impairs the consistency of and causes ambiguity in the AMR algorithm, as is the case in [6].

The key strategy of the present AMR, which relieves us from the necessity of a tree-type data structure, is the proper

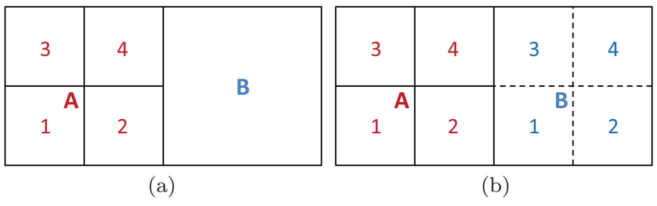


FIG. 2. (Color online) Neighboring blocks at different refinement levels: (a) before refinement and (b) after refinement.

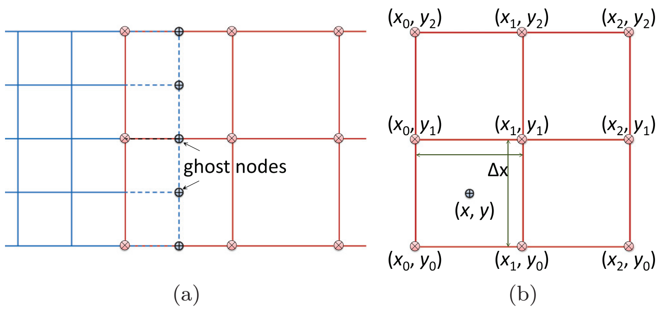


FIG. 3. (Color online) Interblock communications via ghost nodes: blue circles (with a +) are (a) the ghost nodes for the fine block to the left, to be filled from the data points of the coarse block (×) to the right. (b) Interpolation points.

allocation of the child pointers of a newly created block along with a unique designation of the pointers that determine the neighboring blocks. To illustrate this point, let us consider blocks A4 and B in Fig. 2. The east neighbor of block A4 is assigned by associating its pointer with the third child of the east neighbor of its parent, that is, the third child of block B. The FORTRAN pseudocode for this reads as

```
$Block(A4)\%East\_Neighbor$
    $=> Block(A)\%East\_Neighbor\%child(3).$
```

On the other hand, referring to Fig. 2(a), the child pointers for block B, which is still a leaf block with no children, are assigned to their parent ID. Therefore, the east neighbor of block A4 is pointing to block B at this time. Upon refinement of block B in Fig. 2(b), four child blocks are created and their pointers IDs are updated automatically. The east neighbor of block A4 is then pointing to block B3. It is worth noting that the pointers to neighbors are allocated only once upon the creation of the block. After that, no further modification to these pointers is necessary. Once the child IDs are updated, these pointers will automatically point to the active neighboring block.

Now that the connectivity information for interblock and intrablock communications is established, each block should grant access to the portion of the data from the adjacent blocks. This is accomplished by using ghost nodes on each side of the blocks (see Fig. 3). For those neighboring blocks that are at the same refinement level, for example, A1 and A2 or A4 and B3 in Fig. 2(b), the corresponding ghost nodes are easily filled. However, if the neighboring block is one level coarser, as is the case in Fig. 3(a), an interpolation is required. One option would be to use a bilinear interpolation to fill the unavailable data points according to

$$\begin{aligned} f(x,y) = {} & \frac{(x - x_1)(y - y_1)}{(x_0 - x_1)(y_0 - y_1)} f(x_0,y_0) \\ & + \frac{(x_0 - x)(y - y_1)}{(x_0 - x_1)(y_0 - y_1)} f(x_1,y_0) \\ & + \frac{(x - x_1)(y_0 - y)}{(x_0 - x_1)(y_0 - y_1)} f(x_0,y_1) \\ & + \frac{(x_0 - x)(y_0 - y)}{(x_0 - x_1)(y_0 - y_1)} f(x_1,y_1), \end{aligned} \tag{2}$$

which, for the configuration shown in Fig. 3(b), can be simplified to

$$f(x,y) = \frac{f(x_0,y_0)+f(x_1,y_0)+f(x_0,y_1)+f(x_1,y_1)}{4} + O(\Delta x^2). \tag{3}$$

Another option is to use a higher-order interpolation scheme, such as the following biquadratic interpolation written at point (x,y) in Fig. 3(b):

$$\begin{aligned} f(x,y) = [&9f(x_0,y_0)+18f(x_1,y_0)-3f(x_2,y_0)\\ &+18f(x_0,y_1)+36f(x_1,y_1)-6f(x_2,y_1)\\ &-3f(x_0,y_2)-6f(x_1,y_2)+f(x_2,y_2)]/64+O(\Delta x^3). \end{aligned} \tag{4}$$

As will be shown later, not only does the bilinear interpolation lead to less accurate results, but also it causes a pressure jump across the interface of fine-coarse blocks. For the case that a block is adjacent to an external boundary, the corresponding boundary condition is applied instead of filling the ghost nodes.

III. FINITE-DIFFERENCE LATTICE BOLTZMANN METHOD

The discrete Boltzmann equation (DBE) for nearly incompressible single-phase flows [34] with a single-relaxation-time collision operator can be written as

$$\frac{\partial f_\alpha}{\partial t} + \mathbf{e}_\alpha \cdot \nabla f_\alpha = -\frac{f_\alpha - f_\alpha^{eq}}{\lambda}, \tag{5}$$

where f_α is the particle distribution function, t is the time, and λ is the relaxation parameter. For the nine-velocity lattice used in this study, the microscopic velocity set is given by

$$\mathbf{e}_\alpha = \begin{cases} (0,0), & \alpha = 0\\ (\cos\theta_\alpha, \sin\theta_\alpha), & \theta_\alpha = (\alpha-1)\pi/2,\ \alpha = 1\text{–}4\\ \sqrt{2}(\cos\theta_\alpha, \sin\theta_\alpha), & \theta_\alpha = (2\alpha-9)\pi/4,\ \alpha = 5\text{–}8 \end{cases} \tag{6}$$

and the modified equilibrium distribution function is [34]

$$f_\alpha = w_\alpha\left[p + \rho_0 c_s^2\left(\frac{\mathbf{e}_\alpha\cdot\mathbf{u}}{c_s^2} + \frac{(\mathbf{e}_\alpha\cdot\mathbf{u})^2}{2c_s^4} - \frac{(\mathbf{u}\cdot\mathbf{u})}{2c_s^2}\right)\right], \tag{7}$$

where $c_s = 1/\sqrt{3}$ is the lattice speed of sound and the weight coefficients are $w_0 = 4/9$, $w_{1\text{–}4} = 1/9$, and $w_{5\text{–}8} = 1/36$. The density ρ_0 is constant and the pressure and velocity are calculated from the following relations:

$$p = \sum_{\alpha=0}^{8} f_\alpha, \quad \rho_0 c_s^2 \mathbf{u} = \sum_{\alpha=1}^{8} \mathbf{e}_\alpha f_\alpha. \tag{8}$$

Following the idea proposed in [20], we recast the DBE (5) into two separate steps:

collision

$$\hat{f}_\alpha = f_\alpha - \frac{f_\alpha - f_\alpha^{eq}}{\tau + 0.5} \tag{9}$$

and streaming

$$\frac{\partial \hat{f}_\alpha}{\partial t} + \mathbf{e}_\alpha \cdot \nabla \hat{f}_\alpha = 0, \tag{10}$$

where $\tau = \lambda/\Delta t$ is the dimensionless relaxation time and Δt is the time step. Because of the Eulerian nature of the streaming step (10), temporal and spatial discretizations are now decoupled, allowing us to have different spatial resolutions irrespective of the time step. The Taylor-series expansion of the streaming equation is then approximated by

$$\hat{f}_\alpha(\mathbf{x},t+\Delta t) = \hat{f}_\alpha(\mathbf{x},t) - \Delta t\,\mathbf{e}_\alpha\cdot\nabla\hat{f}_\alpha\Big|_{(\mathbf{x},t)} + \frac{\Delta t^2}{2}(\mathbf{e}_\alpha\cdot\nabla)^2\hat{f}_\alpha\Big|_{(\mathbf{x},t)}. \tag{11}$$

Discretizing the first and second derivatives using central differences along characteristic lines leads us to the following Lax-Wendroff scheme, with second-order accuracy in both time and space:

$$\begin{aligned} \hat{f}_\alpha(\mathbf{x},t+\Delta t) = {} & \hat{f}_\alpha(\mathbf{x},t) - \sigma[\hat{f}_\alpha(\mathbf{x},t) - \hat{f}_\alpha(\mathbf{x}-\Delta x_\alpha,t)]\\ &-\tfrac{1}{2}\sigma(1-\sigma)[\hat{f}_\alpha(\mathbf{x}+\Delta x_\alpha,t) - 2\hat{f}_\alpha(\mathbf{x},t)\\ &+\hat{f}_\alpha(\mathbf{x}-\Delta x_\alpha,t)], \end{aligned} \tag{12}$$

where $\sigma = \frac{|\mathbf{e}_\alpha|\Delta t}{\Delta x_\alpha}$ is the CFL number and Δx_α is the grid spacing in the direction of $\mathbf{e}_\alpha$. As can be seen in Eq. (12), when the CFL number is equal to one ($\sigma = 1$) the streaming process collapses into the perfect shift $\hat{f}_\alpha(\mathbf{x},t+\Delta t) = \hat{f}_\alpha(\mathbf{x}-\Delta x_\alpha,t)$, as is the case in the conventional LBM. This occurs when the spatial resolution is at its finest level, for instance, close to the solid walls, which is favorable because this solution is the most accurate one. At other points, where the CFL number is less than one, the streaming step is solved with second-order accuracy in time and space.

Since the collision step is local, we only need to provide boundary conditions for the streaming step, as will be described in the following section. Also, in the present study the collision step is conducted using the multiple-relaxation time (MRT) [35]. This affects only the collision step in Eq. (9) to

$$\hat{f}_\alpha = f_\alpha - \mathbf{M}^{-1}\hat{\mathbf{S}}\mathbf{M}\big(f_\alpha - f_\alpha^{eq}\big), \tag{13}$$

where $\mathbf{M}$ is the orthogonal transformation matrix and $\mathbf{M}^{-1}$ is its inverse [35]. The diagonal relaxation matrix is

$$\hat{\mathbf{S}} = \text{diag}(0,s_e,s_\epsilon,0,s_q,0,s_q,s_\nu,s_\nu). \tag{14}$$

The zero elements in the diagonal relaxation matrix correspond to the conserved quantities (density and momentum in the x and y directions). The relaxation rates related to the energy mode are set to $s_e = s_\epsilon = 1$ and the one related to the energy flux is chosen as $s_q = 1.7$. These relaxation rates are chosen according to linear stability analysis [35,36]. The last two elements are related to the regular relaxation time, and therefore to the kinematic viscosity, by

$$s_\nu = \frac{1}{\tau + 0.5}, \quad \nu = \tau c_s^2. \tag{15}$$

IV. NUMERICAL RESULTS

One of the key components of any AMR algorithm is an error indicator routine to determine whether the mesh in a particular region needs to be refined. It is crucial to have a consistent and reliable estimator to obtain an accurate solution wherever it is needed. Because we are dealing with vortex-dominated flow problems, several refinement criteria in a dimensionless form, based on vorticity, are examined in the following sections.

The magnitude of the dimensionless vorticity is used as the first error indicator:

$$\epsilon_1 = |\omega^*| = \left|\frac{\partial v^*}{\partial x^*} - \frac{\partial u^*}{\partial y^*}\right|, \tag{16}$$

where the variables with an asterisk are dimensionless. The reference velocity U_0 and the length scale L are used to nondimensionalize the vorticity. The vorticity has been used as the refinement criterion in other studies as well (e.g., [9]).

The second indicator is a variation of the relative error in vorticity:

$$\epsilon_2 = \frac{\sqrt{(\Delta_x\omega)^2 + (\Delta_y\omega)^2}}{|\omega|}, \tag{17}$$

where Δ_x and Δ_y are the undivided first derivatives in the x and y directions, respectively. A similar criterion was also used in Refs. [10,11]. This criterion is mostly useful when the mesh close to the walls needs to be refined, as will be seen in the simulation of cavity flow.

The final indicator is based on the so-called Q criterion [37,38]

$$\epsilon_3 = Q = \frac{\|\Omega\|^2}{\|S\|^2}, \tag{18}$$

where S is the rate of strain tensor and Ω is the rate of rotation tensor. These tensors are the symmetric and antisymmetric parts of the velocity gradient tensor, respectively. Here $Q > 1$ indicates regions with larger vortical strength than shear strength.

In this paper, the time step is fixed at $\Delta t = 1$ (all the dimensions are in lattice units). At every time step, the errors are calculated for all the leaf blocks. A block is flagged for refinement if the indicator is above a prescribed value $\epsilon > \epsilon_r$. Similarly, a block is marked for derefinement if $\epsilon < \epsilon_d$. The actual refinement-derefinement is performed by considering the one-level-jump rule for the refinement level of neighboring blocks. In this study, the typical thresholds for refinement and derefinement are, respectively, $\epsilon_r = 0.2/U_0$ and $\epsilon_d = 0.1/U_0$ when using the dimensionless vorticity magnitude ϵ_1, $\epsilon_r = 0.2$ and $\epsilon_d = 0.1$ when using the relative error in vorticity ϵ_2, and $\epsilon_r = \epsilon_d = 1$ when using the Q criterion ϵ_3 as the error indicator.

A. Taylor-Green vortex flow

As a test problem, we consider the Taylor-Green decaying vortex flow in a periodic domain with the following analytic solution [36,39,40]:

$$\begin{aligned} u_a(x,y) &= -U_0\cos(kx)\sin(ky)\exp(-t/t_c),\\ v_a(x,y) &= U_0\sin(kx)\cos(ky)\exp(-t/t_c), \\ p_a(x,y) &= p_0 - \frac{\rho_0 U_0^2}{4}[\cos(2kx)+\cos(2ky)]\exp(-2t/t_c), \end{aligned} \tag{19}$$

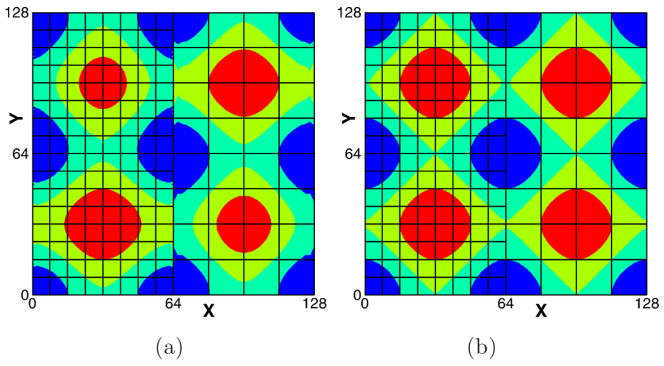


FIG. 4. (Color online) Pressure contour in decaying vortex flow: comparison between (a) bilinear and (b) biquadratic interpolations.

where $k = \frac{2\pi}{L}$ is the wave number and $t_c = \frac{1}{2\nu k^2}$ is the viscous time. The constant parameter p_0 is set to zero and the initial conditions are set by calculating Eq. (19) at $t = 0$.

In order to assess the accuracy of the proposed model, three systematically refined grids are considered at Re = 20. The Mach number of the flow Ma $= U_0/c_s$ is kept below 0.017. For each case, the left half of the domain is covered with blocks with $\frac{\Delta x_\alpha}{|\boldsymbol{e}_\alpha|} = 1$, while the right half of the domain consists of blocks with $\frac{\Delta x_\alpha}{|\boldsymbol{e}_\alpha|} = 2$ (see Fig. 4). This has two imminent implications. First, the conventional perfect shift is implied as the solution to the streaming equation (10) on the left side of the domain, while the Lax-Wendroff scheme with $\sigma = 0.5$ gives the solution on the right side. Second, at the interface between the fine-coarse blocks in the middle of the domain, an interpolation is necessary to provide the distribution functions needed to perform the streaming step on the finer blocks. This gives us the opportunity to examine the effect of interpolation schemes on the order of accuracy of the model as well.

The computations are done for three different values of maximum refinement level $l_{\max} = 3$, 4, and 5, which correspond to the domain lengths of $L = 64, 128$, and 256, respectively. The numerical results are compared to the analytical solution given by Eq. (19) at $t = t_c$ and the error is measured by calculating the L_2 norm

$$\|\delta \boldsymbol{u}\|_2 = \sqrt{\frac{\sum_{x,y}[(u-u_a)^2 + (v-v_a)^2]}{\sum_{x,y}\left(u_a^2 + v_a^2\right)}}, \tag{20}$$

where u and v are the macroscopic velocities computed using FDLBM and the summation is taken over the entire domain.

As can be seen in Fig. 5, bilinear interpolation degrades the order of accuracy of the FDLBM to first order. This is due to the fact that, although bilinear interpolation gives us a second-order-accurate value for the distribution function, it causes the gradient calculation of the distribution function in the streaming step (10) to be of first-order accuracy, thus reducing the overall accuracy of the FDLBM to first order. Comparing the bilinear and biquadratic interpolations in Fig. 5, we observe that the bilinear interpolation not only makes the FDLBM first order, but also produces an error that is about one order of magnitude larger than that of the biquadratic interpolation. Moreover, as shown in Fig. 4,

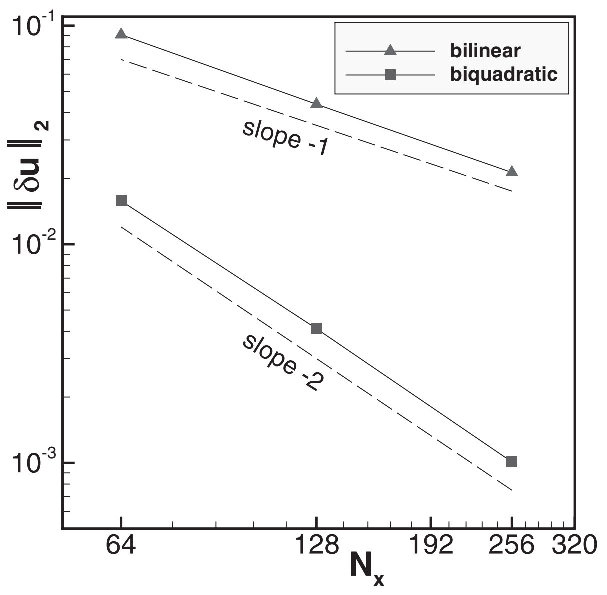


FIG. 5. Convergence test: order of accuracy of the proposed model using bilinear and biquadratic interpolations.

bilinear interpolation causes a discontinuity in the pressure at the border of fine-coarse blocks. This behavior was also observed by Lagrava *et al.* [28]. It is worth mentioning, however, that when we adaptively refine-derefine the mesh, this pressure jump across the interface of fine-coarse blocks is greatly suppressed. Nevertheless, the bilinear interpolation should be used with great care.

B. Lid-driven cavity flow

A lid-driven cavity flow is simulated to further validate and evaluate the accuracy of the code with regard to the error indicators introduced earlier. The Reynolds number of the flow $\mathrm{Re} = U_0 L/\nu$ is set to 3200. The lid of the cavity, at the top, is moving to the right with $U_0 = 0.1$ ($\mathrm{Ma} = 0.17$). Figure 6 shows the streamlines for three different error criteria along with the streamlines on a uniform mesh after 200 000 iterations. Qualitatively, the first and the second error criteria, defined in Eqs. (16) and (17), seem to be more successful in capturing the large eddies at the corners of the cavity. The third error criterion, however, has failed to capture the eddies, especially the smaller one at the top left corner, as elegantly as the other two indicators. The arrangement of the blocks for all the refinement criteria can be seen in Fig. 7. It is interesting to note that, although the first two plots in Fig. 7 have substantially different configurations, they result in almost the same solution, qualitatively. On the other hand, the performance of the last estimator, based on the Q criterion, is very poor in detecting the large gradient regions.

In order to quantify the errors associated with these refinement indicators, we calculate the difference between the velocity fields based on the uniform mesh and the AMR solutions. The absolute difference in the magnitude of the velocity and the L_2 norm are presented in Table I. The common points for different block configurations in Fig. 7 are used to calculate the errors. As can be seen in Table I, the errors given for the last refinement criterion defined in Eq. (18) is one order

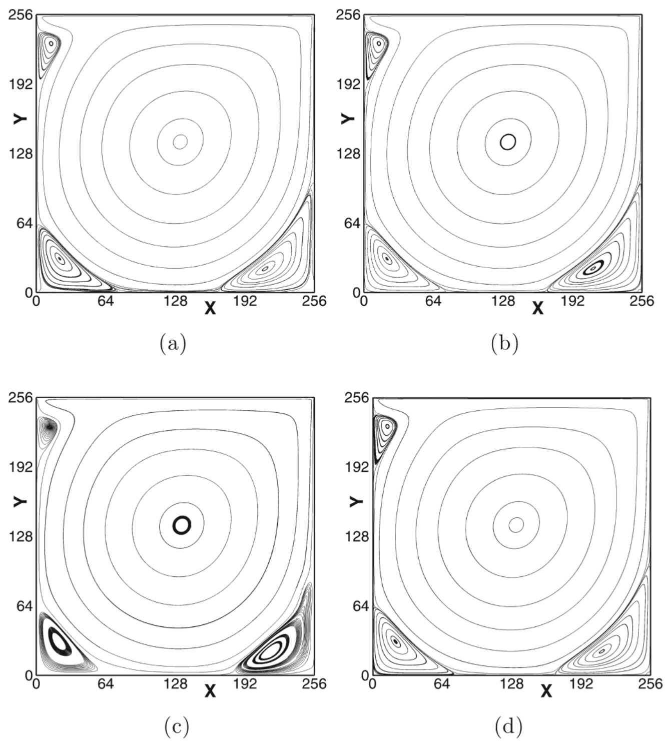


FIG. 6. Lid-driven cavity flow at Re = 3200 after 200 000 iterations. Comparison of different error indicators: (a) ϵ_1, (b) ϵ_2, (c) ϵ_3, and (d) uniform mesh.

of magnitude larger than the other two indicators. Although the first error indicator, which is based on the vorticity, gives us acceptable results with less than 0.5% discrepancy with uniform-mesh results, the second refinement criterion, defined in Eq. (17), gives us the smallest error (0.3%) for this particular problem. This is due to the fact that all the blocks adjacent to the wall boundaries are refined to their finest levels as a result of using a relative error in vorticity.

We also compared the FDLBM AMR results to the uniform-mesh solution and the benchmark study by Ghia *et al.* [41]. In Fig. 8, the centerline velocities inside the lid-driven cavity flow at Re = 3200 are plotted and satisfactory agreement with the benchmark study is observed. The very small difference between the results is due to the fact that Ghia *et al.* [41] used a moderate 128 × 128 grid, while we are using a 256 × 256 grid for the uniform-mesh solution.

C. Thin shear layer

In a recent study [42], we examined the instability of a shear layer for single-phase and multiphase flows using an

TABLE I. Absolute and relative errors of the velocity field in the simulation of cavity flow using different error criteria (the uniform-mesh result is used as the reference solution).

Refinement criterion	$\sum_{x,y} \lvert\Delta \boldsymbol{u}\rvert$	$\lVert\delta \boldsymbol{u}\rVert_2$
ϵ_1 [Eq. (16)]	0.030	0.0049
ϵ_2 [Eq. (17)]	0.018	0.0029
ϵ_3 [Eq. (18)]	0.336	0.0538

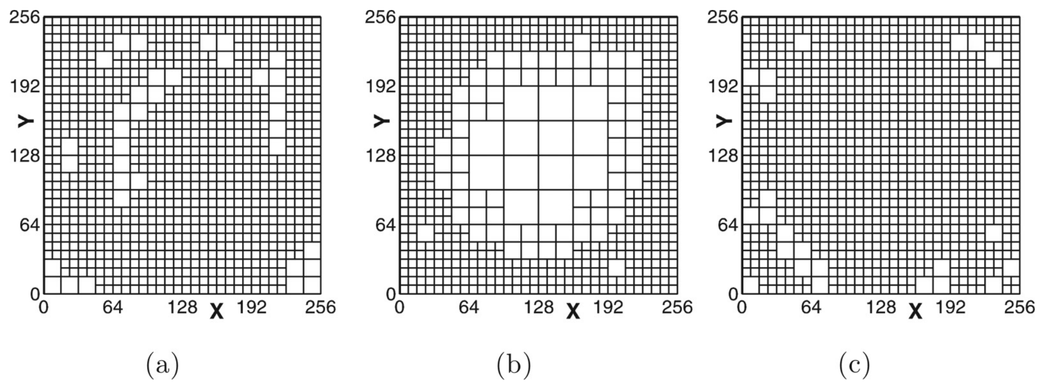


FIG. 7. Configuration of the adaptively created blocks in the cavity flow for different error indicators: (a) ϵ_1, (b) ϵ_2, and (c) ϵ_3.

MRT LBM on a uniform mesh. In this section, the stability of a single-phase Kelvin-Helmholtz billow is considered and the results are compared to the findings of Fontane and Joly [43], who solved the Navier-Stokes equations using an AMR technique to study the Kelvin-Helmholtz instability of two miscible fluids in the absence of gravity and interfacial tension. The following velocity profile, which consists of a base flow and small perturbations for the initiation of the Kelvin-Helmholtz instability, is imposed across the shear layer:

$$\begin{aligned} u(y) &= U_0 \tanh\left(\frac{y}{\delta}\right) + \varepsilon u'(y)\cos(\alpha x), \\ v(y) &= \varepsilon v'(y)\cos(\alpha x), \end{aligned} \tag{21}$$

where δ is half the shear-layer depth or vorticity thickness, $\alpha = 2\pi/\lambda$ is the wave number with λ being the wavelength, ε is the perturbation amplitude, and u' and v' are complex eigenfunctions of the Rayleigh stability equation [43,44]. For the most unstable mode we found $\alpha\delta = 0.444\,92$ [42].

We seed the initial disturbances similarly to what was performed in [43]. The size of the computation domain corresponds to a uniform mesh with 256×256 lattice points. The boundary conditions are periodic in the streamwise direction and free slip at the top and bottom. The Reynolds number of the flow is

$$\text{Re} = \frac{U_0\delta}{\nu}, \tag{22}$$

where U_0 is half the velocity difference across the shear layer. Here $U_0 = 0.1$, which results in $\text{Ma} = 0.17$. The dimensionless time is also defined by $t^* = tU_0/\delta$.

A comparison of the performance of the error estimators is presented in Fig. 9. While the best result is obtained using the vorticity magnitude as the error indicator in Fig. 9(a), the error criterion based on the relative error in vorticity does not yield very accurate results. This is contrary to what we observed in the cavity flow simulations. The main reason is that we are dealing with a larger vorticity value in the middle of the shear layer compared to the small eddies at the corners of the cavity. Because the vorticity magnitude appears in the denominator of Eq. (17), the corners of the cavity, where the magnitude of the vorticity was small, were flagged for refinement. However, in the middle of the shear layer, the magnitude of the vorticity is rather large and the difference in vorticity is small. Therefore, the middle part of the domain was not flagged for refinement. The opposite is the case for the parts of the shear layer that are far from the center; the vorticity magnitude is close to zero, making the denominator of Eq. (17) small, and incorrectly signals to refine the mesh away from the center. Consequently, the results shown in Fig. 9(b) are not in good agreement with the uniform-mesh solution. Once again, the results of the Q-criterion indicator, depicted in Fig. 9(c), are not in line with the uniform-mesh results.

We further verify our explanations, which are based on qualitative comparisons, by quantifying the errors between AMR results and uniform-mesh results. Once again, the absolute error in the velocity field as well as the L_2 norm is calculated and the results are given in Table II. As can be

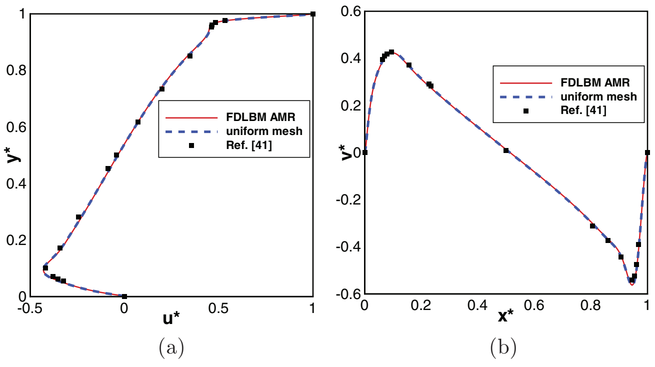


FIG. 8. (Color online) Centerline velocities inside the cavity flow at Re = 3200: (a) dimensionless x velocity along the vertical centerline and (b) dimensionless y velocity along the horizontal centerline.

TABLE II. Absolute and relative errors of the velocity field in the simulation of thin layer shear flow using different error criteria (the uniform-mesh result is used as the reference solution).

Refinement criterion	$\sum_{x,y} \lvert\Delta \boldsymbol{u}\rvert$	$\lVert\delta \boldsymbol{u}\rVert_2$
ϵ_1 [Eq. (16)]	0.019	0.0014
ϵ_2 [Eq. (17)]	0.094	0.0067
ϵ_3 [Eq. (18)]	0.362	0.0259

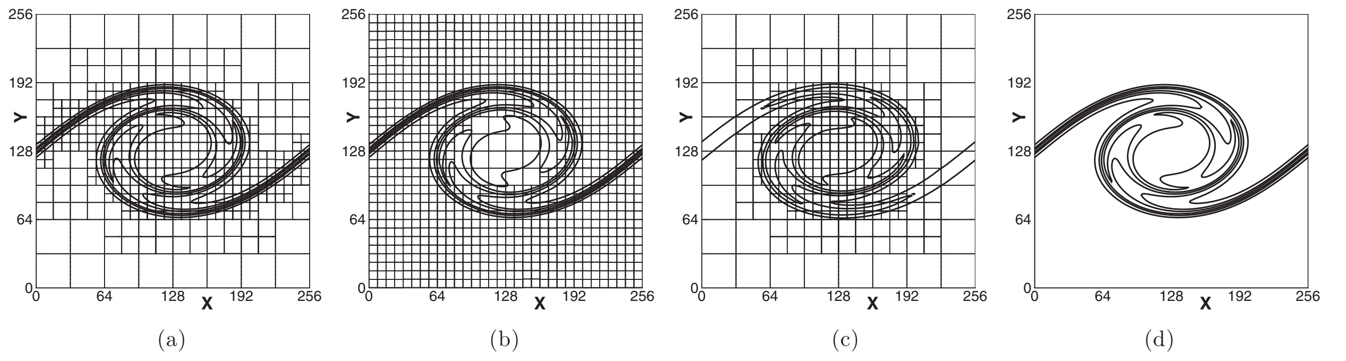


FIG. 9. Vorticity contour levels for Kelvin-Helmholtz instability at Re = 1500 and $t^* = 26$. Comparison of different estimators: (a) ϵ_1, (b) ϵ_2, (c) ϵ_3, and (d) uniform mesh. The vorticity increment between contours is 1/6.

seen in Table II, the first indicator defined in Eq. (16) gives us the most accurate results with about 0.1% discrepancy with the uniform-mesh solution. The error for the second criterion, although acceptable, is 5 times larger than the first one. On the other hand, the Q criterion gives us the least accurate results with more than 2% error.

A comparison of our FDLBM AMR result to that of Fontane and Joly [43] is also shown in Fig. 10. As can be seen, the current FDLBM AMR results, using the magnitude of the dimensionless vorticity as the error indicator, are in good agreement with those obtained in [43].

From the obtained results, we see that the choice of the error criterion is rather case specific. When dealing with stationary walls, the error indicator based on the relative difference in vorticity helps to refine the mesh close to the wall boundaries. In the absence of exterior walls, however, the magnitude of the vorticity is found to be a reliable estimator in detecting gradients and capturing the essential physics of the problem. Of course, the identification of a universal error indicator for AMR schemes is still the subject of ongoing research and will not be further discussed in this study.

D. Flow past a square cylinder

Although the AMR technique can be applied to steady state flows, its maximum benefit is obtained when it is used for the simulation of unsteady flows with large variations in the scale of the physics of the problem. Moreover, the versatility of an AMR scheme is better manifested when applied to unsteady flows such as the vortex-shedding phenomenon behind a cylinder. The physics of a uniform flow past a square cylinder placed in an infinite domain has been studied extensively both numerically and experimentally [45–51]. There are also a few studies on the flow pattern behind a square box in a confined channel (with blockage effects) using the LBM [52–54]. For a uniform flow over a square cylinder, it is well known that once the Reynolds number is above a critical value $\mathrm{Re}_{cr} \sim 52$ [47,52], the flow field becomes unstable and the so-called von Kármán vortex street is generated downstream of the flow. To examine the accuracy of the present FDLBM AMR, the unsteady flow over a square cylinder in an unbounded uniform flow (without blockage effects) is examined.

The computational domain is chosen similar to previous studies [47–49], as depicted in Fig. 11. A square box with length D is placed at $x = 8D$ from the inlet. The length and height of the domain are equal to $L = 32D$ and $H = 16D$, respectively. The no-slip boundary condition is applied at the walls of the square cylinder, while a uniform flow profile $(u,v) = (U_0,0)$ is imposed at the inlet, top, and bottom of the domain, as shown in Fig. 11. At the outlet, the pressure is fixed and the following convective boundary condition is solved to update the velocity:

$$\frac{\partial \mathbf{u}}{\partial t} + U_0 \frac{\partial \mathbf{u}}{\partial x} = 0. \tag{23}$$

The far-field boundary conditions are imposed using the momentum-exchange scheme [15] at the inlet, top, and bottom boundaries, while the distribution function is set equal to the equilibrium value at the outlet. On the walls of the square cylinder, the second-order link bounceback is applied. The initial condition is a constant pressure with a uniform velocity

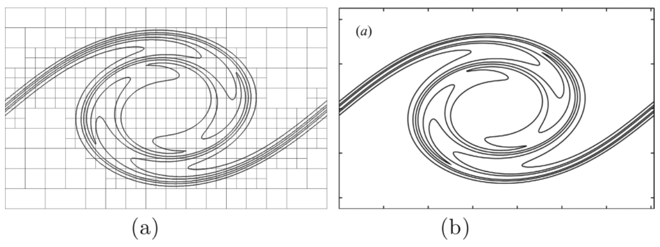


FIG. 10. Comparison of (a) current FDLBM AMR findings with (b) AMR benchmark results in Ref. [43] [Fig. 2(a) therein] for Kelvin-Helmholtz instability at Re = 1500 and $t^* = 26$ with $l_{\max} = 5$.

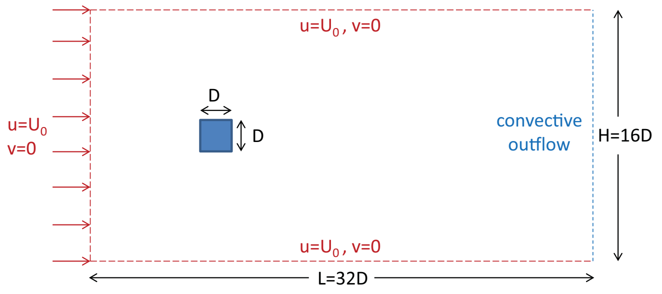
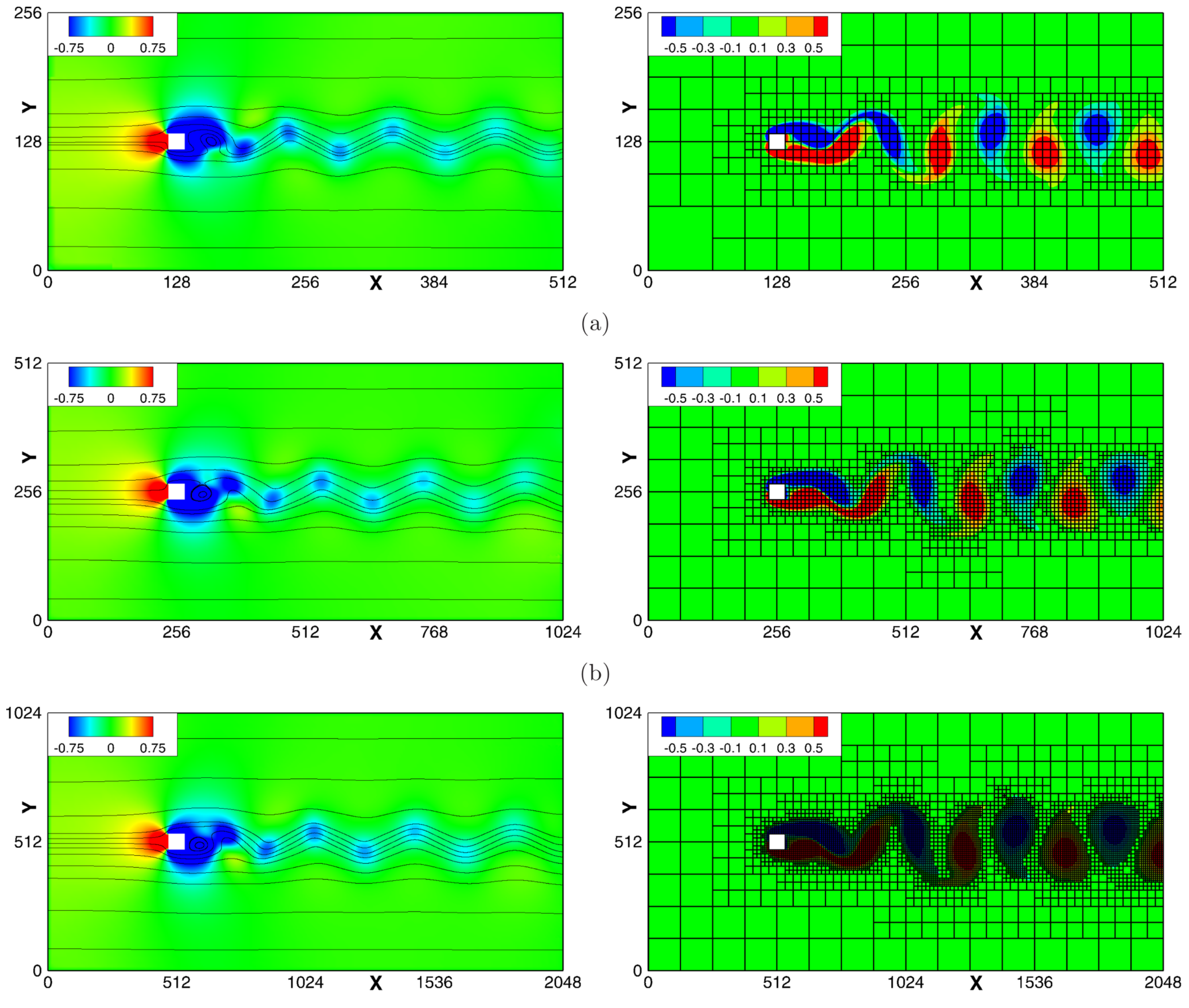


FIG. 11. (Color online) Flow past a square cylinder: computational domain and boundary conditions.



(c)

FIG. 12. (Color online) Vortex shedding past a square cylinder at Re = 100 for (a) $l_{\max} = 5$ ($D = 16$), (b) $l_{\max} = 6$ ($D = 32$), and (c) $l_{\max} = 7$ ($D = 64$). Shown on the left are streamlines and pressure contours and on the right are vorticity contours and mesh configurations.

profile everywhere except in the vicinity of the square cylinder, where the velocity is set to zero. The important parameters of the flow field are the Reynolds number and the Strouhal number, which are defined as

$$\mathrm{Re} = \frac{U_0 D}{\nu}, \qquad \mathrm{St} = \frac{f_s D}{U_0}, \tag{24}$$

where f_s is the shedding frequency determined from the alternation of the lift force at the surface of the cylinder. The momentum-exchange method [15] is used to calculate the force exerted on the cylinder:

$$\mathbf{F} = \sum_{x_b} \sum_{\alpha=1}^{8} \mathbf{e}_\alpha [f_\alpha(\mathbf{x}_b,t) + f_\beta(\mathbf{x}_b + \Delta x_\alpha,t)]. \tag{25}$$

For bounceback on the link, the effective location of the wall is halfway between the boundary nodes and the fluid nodes. As a result, the momentum exchange between the cylinder and the flow field can be rewritten as

$$\mathbf{F} = 2 \sum_{x_b} \sum_{\alpha=1}^{8} \mathbf{e}_\alpha f_\alpha(\mathbf{x}_b,t). \tag{26}$$

After evaluating the force on the cylinder, the drag and lift coefficients are calculated by

$$C_D = \frac{2F_x}{\rho_0 U_0^2 D}, \qquad C_L = \frac{2F_y}{\rho_0 U_0^2 D}. \tag{27}$$

Note that the actual value of D used in Eqs. (24) and (27) is the input value (16, 32, or 64) plus one. The one comes from the fact that, as a result of the bounceback on the link, the actual

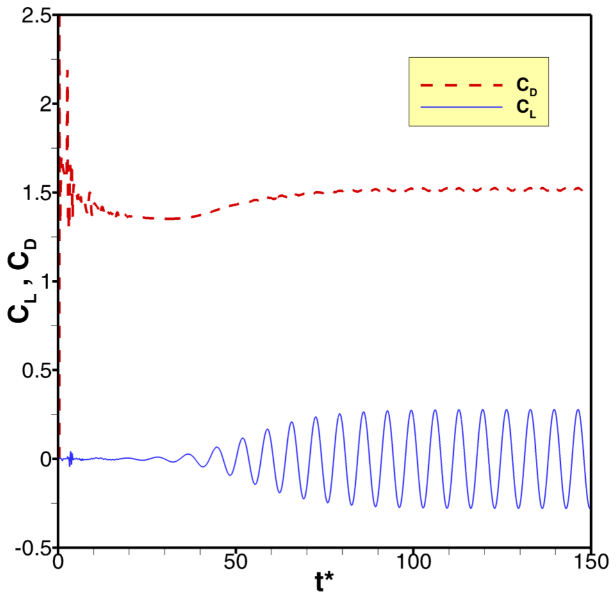


FIG. 13. (Color online) Drag and lift coefficients versus dimensionless time for a uniform flow past a square cylinder at Re = 100.

location of the wall is halfway between the fluid nodes and the solid nodes.

The vortex-shedding phenomenon past a square cylinder is studied at Re = 100 (Ma = 0.17) and the numerical values of the time-averaged drag coefficient and the Strouhal number are measured. The grid consistency of the results is ensured using three different grid resolutions around the square cylinder. The sides of the square cylinder are resolved using 16, 32, and 64 lattice nodes, corresponding to $l_{\max} = 5$, 6, and 7, respectively. In Table III, the obtained results are compared with previous studies. As can be seen, the calculated drag coefficient and the Strouhal number are in reasonable agreement with the benchmark studies. Also, increasing the resolution from $l_{\max} = 6$ ($D = 32$) to $l_{\max} = 7$ ($D = 64$) does not affect the predicted values for the drag and the Strouhal number, confirming the grid convergence of the findings.

The contour plots of pressure and streamlines, as well as the dynamically created blocks, are shown in Fig. 12 for $l_{\max} = 5$–7. As can be seen, the refinement criterion based on the vorticity magnitude, defined in Eq. (18), is fairly successful in capturing the von Kármán vortex structures. The mesh is refined where it needs to be, producing a more accurate

TABLE III. Drag coefficient and Strouhal number for unsteady flow past a square cylinder at Re = 100.

Reference	$\bar{C}_D$	St
[46]		0.141–0.145
[47]	1.44–1.48	0.144–0.146
[48]	1.51	0.159
[49]	1.51	0.150
[50]	1.40–1.53	0.144–0.146
present work ($D = 16$)	1.53	0.151
present work ($D = 32$)	1.51	0.150
present work ($D = 64$)	1.51	0.149

TABLE IV. Efficiency of the proposed FDLBM AMR compared to a uniform-mesh solver.

Case study	Uniform-mesh time (s)	FDLBM AMR time (s)	Speedup
cavity flow	2678	2957	0.9
thin shear layer	65	38	1.7
vortex shedding ($l_{\max} = 5$)	701	331	2.1
vortex shedding ($l_{\max} = 6$)	6074	1827	3.3
vortex shedding ($l_{\max} = 7$)	46652	12081	3.9

solution in the regions of interest. Also, Fig. 13 plots the variation of drag and lift coefficients versus dimensionless time for the square cylinder with $D = 64$ lattice nodes ($l_{\max} = 7$).

Finally, one complete cycle of the von Kármán vortex street behind the square cylinder at Re = 100 is illustrated in Fig. 14. The interval between subsequent snapshots is one-fourth of the vortex-shedding period. Fluctuations in the wake velocity are clearly seen in the vorticity contours downstream of the flow.

E. Efficiency

In this section we compare the CPU time of our FDLBM AMR code with the uniform-mesh code. All the simulations are performed on a PC with Intel(R) Core(TM) i7 2.8 GHz CPU and 4 GB RAM. The full optimization option is used. It is worth noting that in all the data reported in Table IV, the refinement-derefinement subroutine, which costs about 6% of the computation time, is called at each time step.

As the results presented in Table IV suggest, the AMR strategy is not very helpful when dealing with the flow in a confined geometry like the cavity flow. This is due to the fact that sharp gradients close to wall boundaries causes the mesh to be refined in those regions. As illustrated in the cavity flow simulations in Fig. 7, a high percentage of the computational domain is refined to its finest level degrading the efficiency of an AMR routine. On the other hand, thin shear layer and vortex-shedding simulations show promising results in the speedup factor when using the AMR routine. The efficiency of the FDLBM-AMR increases with the maximum refinement level.

V. SUMMARY

An adaptive mesh refinement strategy in the framework of the lattice Boltzmann method was developed in this study. Compared to available AMR structures, the proposed AMR is easier to code and implement mainly because the modification of a tree-type data structure after the reconstruction step is no longer needed. Meanwhile, there is no need to search a tree structure [3] or look up entries in a hash table [6] to find the neighbors of a given block because the neighboring blocks are already known via using pointers. Also, in the absence of a tree structure, executing the code in parallel is intrinsically straightforward.

Using the AMR technique, a FDLBM on nonuniform grids was presented that avoids temporal interpolations at the borders of fine-coarse grids by invoking a unified time step throughout the computational domain. This has two immediate advantages: First, the second-order accuracy of the LBM in

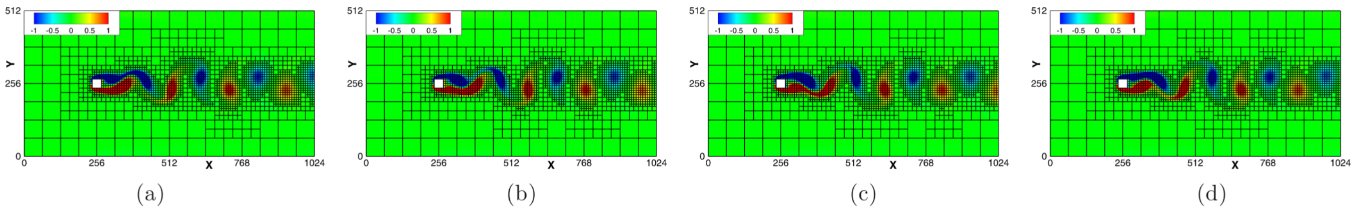


FIG. 14. (Color online) Vorticity contours past a square cylinder at Re $= 100$ with $l_{\max} = 6$ for (a) t_0, (b) $t_0 + \frac{1}{4f_s}$, (c) $t_0 + \frac{2}{4f_s}$, and (d) $t_0 + \frac{3}{4f_s}$.

time is retained and second, there is no need to rescale the distribution functions at the borders of fine-coarse grids. In the meantime, we take advantage of the perfect shift in the streaming process, as is the case in the conventional LBM, on the blocks that are at the finest refinement level where the most accurate results are demanded.

Comparison of the accuracy and efficiency of the current FDLBM with the conventional stream-and-collide LBM on nonuniform grids is beyond the scope of the present paper. Also, extension of the proposed FDLBM AMR to multiphase flows is a subject left for future work.

ACKNOWLEDGMENT

This research was carried out using funding received from the DOE's NEUP.

[1] M. J. Berger and J. Oliger, J. Comput. Phys. **53**, 484 (1984).
[2] M. J. Berger and P. Colella, J. Comput. Phys. **82**, 64 (1989).
[3] D. de Zeeuw and K. G. Powell, J. Comput. Phys. **104**, 56 (1993).
[4] A. M. Khokhlov, J. Comput. Phys. **143**, 519 (1998).
[5] A. V. Kravtsov, A. A. Klypin, and A. M. Khokhlov, Astrophys. J. Suppl. Ser. **111**, 73 (1997).
[6] H. Ji, F.-S. Lien, and E. Yee, J. Comput. Phys. **229**, 8981 (2010).
[7] P. MacNeice, K. Olson, C. Mobarry, R. de Fainchtein, and C. Packer, Comput. Phys. Commun. **126**, 330 (2000).
[8] R. Teyssier, Astron. Astrophys. **385**, 337 (2002).
[9] S. Popinet, J. Comput. Phys. **190**, 572 (2003).
[10] T. Matsumoto, Publ. Astron. Soc. Jpn. **59**, 905 (2007).
[11] U. Ziegler, Comput. Phys. Commun. **179**, 227 (2008).
[12] R.-L. Jiang, C. Fang, and P.-F. Chen, Comput. Phys. Commun. **183**, 1617 (2012).
[13] J. Rantakokko and M. Thun, *Parallel Computing* (Springer, London, 2009), Chap. 5.
[14] S. Chen and G. Doolen, Annu. Rev. Fluid Mech. **30**, 329 (1998).
[15] D. Yu, R. Mei, L.-S. Luo, and W. Shyy, Prog. Aerosp. Sci. **39**, 329 (2003).
[16] X. He, L.-S. Luo, and M. Dembo, J. Comput. Phys. **129**, 357 (1996).
[17] D. Yu, R. Mei, and W. Shyy, Int. J. Numer. Methods Fluids **39**, 99 (2002).
[18] D. Kandhai, W. Soll, S. Chen, A. Hoekstra, and P. Sloot, Comput. Phys. Commun. **129**, 100 (2000).
[19] V. Sofonea and R. F. Sekerka, J. Comput. Phys. **184**, 422 (2003).
[20] T. Lee and C.-L. Lin, J. Comput. Phys. **185**, 445 (2003).
[21] F. Nannelli and S. Succi, J. Stat. Phys. **68**, 401 (1992).
[22] H. Chen, Phys. Rev. E **58**, 3955 (1998).
[23] M. Rohde, D. Kandhai, J. J. Derksen, and H. E. A. van den Akker, Int. J. Numer. Methods Fluids **51**, 439 (2006).
[24] T. Lee and C.-L. Lin, J. Comput. Phys. **171**, 336 (2001).
[25] O. Filippova and D. Hanel, J. Comput. Phys. **147**, 219 (1998).
[26] O. Filippova and D. Hanel, J. Comput. Phys. **165**, 407 (2000).
[27] C.-L. Lin and Y.-G. Lai, Phys. Rev. E **62**, 2219 (2000).
[28] D. Lagrava, O. Malaspinas, J. Latt, and B. Chopard, J. Comput. Phys. **231**, 4808 (2012).
[29] J. Tolke, S. Freudiger, and M. Krafczyk, Comput. Fluids **35**, 820 (2006).
[30] Z. Yu and L.-S. Fan, J. Comput. Phys. **228**, 6456 (2009).
[31] J. Wu and C. Shu, J. Comput. Phys. **230**, 2246 (2011).
[32] Y. Chen, Q. Kang, Q. Cai, and D. Zhang, Phys. Rev. E **83**, 026707 (2011).
[33] G. Eitel-Amor, M. Meinke, and W. Schroder, Comput. Fluids **75**, 127 (2013).
[34] X. He and L.-S. Luo, J. Stat. Phys. **88**, 927 (1997).
[35] P. Lallemand and L.-S. Luo, Phys. Rev. E **61**, 6546 (2000).
[36] R. Mei, L.-S. Luo, P. Lallemand, and D. d'Humières, Comput. Fluids **35**, 855 (2006).
[37] P. Chakraborty, S. Balachandar, and R. J. Adrian, Annu. Rev. Fluid Mech. **535**, 189 (2005).
[38] S. J. Kamkar, A. M. Wissink, V. Sankaran, and A. Jameson, J. Comput. Phys. **230**, 6271 (2011).
[39] A. Bardow, I. V. Karlin, and A. A. Gusev, Phys. Rev. E **77**, 025701 (2008).
[40] T. Kruger, F. Varnik, and D. Raabe, Phys. Rev. E **82**, 025701 (2010).
[41] U. Ghia, K. N. Ghia, and C. T. Shin, J. Comput. Phys. **48**, 387 (1982).
[42] A. Fakhari and T. Lee, Phys. Rev. E **87**, 023304 (2013).
[43] J. Fontane and L. Joly, J. Fluid Mech. **612**, 237 (2008).
[44] A. Michalke, J. Fluid Mech. **19**, 543 (1964).
[45] R. W. Davis and E. F. Moore, J. Fluid Mech. **116**, 475 (1982).
[46] A. Okajima, J. Fluid Mech. **123**, 379 (1982).
[47] A. Sohankar, C. Norberg, and L. Davidson, Int. J. Numer. Methods Fluids **26**, 39 (1998).
[48] A. K. Saha, K. Muralidhar, and G. Biswas, J. Eng. Mech. **126**, 523 (2000).

[49] A. N. Pavlov, S. S. Sazhin, R. P. Fedorenko, and M. R. Heikal, Int. J. Numer. Methods H. **10**, 6 (2000).
[50] N. Hasan, S. F. Anwer, and S. Sanghi, J. Comput. Phys. **206**, 661 (2005).
[51] S. Ul-Islam, C. Y. Zhou, A. Shah, and P. Xie, J. Mech. Sci. Technol. **26**, 1027 (2012).
[52] M. Breuer, J. Bernsdorf, T. Zeiser, and F. Durst, Int. J. Heat Fluid Flow **21**, 186 (2000).
[53] S. Izquierdo, P. Martínez-Lera, and N. Fueyo, Comput. Math. Appl. **58**, 914 (2009).
[54] D. A. Perumal, G. V. S. Kumar, and A. K. Dass, ISRN Math. Phys. **2012**, 630801 (2012).