Journal of Computational Physics **181,** 675–704 (2002)
doi:10.1006/jcph.2002.7151

Large Eddy Simulations of a Stirred Tank Using the Lattice Boltzmann Method on a Nonuniform Grid

Zhenyu Lu,* Ying Liao,* Dongying Qian,* J. B. McLaughlin,*
J. J. Derksen,† and K. Kontomaris‡

*Department of Chemical Engineering, Clarkson University, Potsdam, New York 13699-5705; †Kramers Laboratorium voor Fysische Technologie, Delft University of Technology, 2628 Bw Delft, The Netherlands; and ‡Dupont Central Research & Development, Wilmington, Delaware 19880-0249
E-mail: jmclau@clarkson.edu

Received August 13, 2001; revised May 22, 2002

A nonuniform grid lattice Boltzmann technique previously described by He *et al.* [1] has been extended to simulate three-dimensional flows in complex geometries. The technique is applied to the computation of the turbulent flow in a stirred tank driven by a standard Rushton turbine. With the nonuniform grid approach, the total CPU time required for a simulation of the flow in a stirred tank can be reduced by roughly 75% and still provide the same spatial accuracy as would be obtained with a uniform high-resolution grid. Statistical results for the computed flow fields will be compared with experimental results (H. Wu and G. K. Patterson, *Chem. Eng. Sci.* **44**, 2207 (1989)) and with simulations by J. G. M. Eggels (*Int. J. Heat Fluid Flow* **17**, 307 (1996)) and J. J. Derksen and H. E. A. Van den Akker (*AIChE J.* **45**, 209 (1999)). The results of the nonuniform mesh simulation are in reasonable agreement with the available experimental data and the results of previous simulations.

Key Words: lattice Boltzmann method; grid refinement; stirred tank; turbulence.

1. INTRODUCTION

Eggels [3] presented the results of large eddy simulations (LES) of a turbulent stirred tank flow using the lattice Boltzmann method (LBM). He showed phase-averaged velocity measurements and snapshots of the flow field in the tank. Derksen and Van den Akker [4] made a more detailed study using a similar approach. They collected statistics not only for the phase-averaged results but also for the phase-resolved data that can reveal the trailing vortex structures. They also presented the contours of the turbulent kinetic energy

0021-9991/02 $35.00

and the phase-averaged energy dissipation rate. Comparisons were made between their phase-resolved results and LDA experimental data by Wu and Patterson [2].

Both the simulations and the experimental measurements reveal that the turbulence in the tank is highly inhomogeneous with the largest turbulence dissipation rates confined to a relatively small region near the impeller. These findings suggest that one could reduce the very large computation times needed for LES of stirred tanks by using nonuniform grids with the resolution concentrated in the regions of the tank where the turbulence is most intense. This motivated the present study, which involves an extension of previous work on nonuniform grids to the simulation of the time-dependent, three-dimensional turbulent flow in a stirred tank.

Several researchers have extended the LBM to nonuniform lattices. Nannelli and Succi [5] and Amati *et al*. [6] developed a finite-volume version of the LBM (FVLBM) that involves a nonuniform coarse lattice that contains several of the basic lattice cells. Filippova and Hänel [7] proposed a local grid refinement method based on the hierarchical grid refinement in conventional CFD methods. He *et al*. [1] proposed a method that He and Doolen [8] called the "interpolation-supplemented lattice Boltzmann equation model" (ISLBE). This method applies interpolation to update the particle distribution functions on the coarse portion of the grid after the propagation step. He *et al*. applied the ISLBE method to simulate the flow in a 2D symmetric channel with a sudden expansion on a nonuniform rectangular grid. Since the grid was coarse in one direction, one-dimensional quadratic interpolation was used to obtain a result with second-order accuracy. Later, He and Doolen [8] constructed a curvilinear coordinate system by applying the ISLBE method to simulate the vortex shedding behind a circular cylinder. Two-dimensional second-order upwind interpolation was applied in their simulation. Compared with the other nonuniform grid techniques, the ISLBE method is easy for programming, parallel computation, and incorporation of microscopic interactions in multiphase flow computations. It is valid for grids with arbitrary shapes and for both the BGK scheme and the Somers–Eggels scheme.

In the work reported in this paper, a nonuniform grid technique developed from the ISLBE method is applied to improve the efficiency of the large eddy simulation of a stirred tank; the LDA experimental results reported by Wu and Patterson [2] and Derksen *et al*. [9, 10] will be compared with the simulation results.

The nonuniform LBM simulations in this paper have several features that were not present in the previous ISLBE method. The simulations to be reported were performed on a three-dimensional FCHC lattice having 18 directions. A second difference is that the ISLBE scheme presented by He *et al*. [1] applies interpolation after the propagation step to redistribute the particle distribution functions, while the new scheme replaces the propagation step with interpolation. This results in a further improvement in computational efficiency. Finally, the computations are eliminated on grid points that lie outside the computational domain using an approach similar to that employed by Eggels [3] for uniform grids. It will be shown that in addition to reducing the memory requirements by roughly 65%, the total CPU time can be reduced by roughly 75% in the large eddy simulation of a stirred tank without significant sacrifices in accuracy.

The paper is organized as follows. Section 2 presents a concise review of the Somers–Eggels LBM scheme. The nonuniform grid technique is presented in Section 3. The implementation of boundary conditions with the nonuniform grid method is discussed in Section 4. Section 5 discusses the parallelization of the code. Section 6 discusses an "arbitrary computational domain" method in which the computations are skipped for points

lying outside the domain of physical interest. The simulation procedures for a stirred tank are presented in Section 7. The results of the simulations are presented and discussed in Section 8. Finally, the conclusions are presented in Section 9.

2. SOMERS–EGGELS LBM SCHEME

The lattice Boltzmann method (LBM) is a relatively new type of time-dependent Navier–Stokes solver in the incompressible flow regime. The fundamental idea of the LBM is to use a microscopic model of a many-particle system obeying the conservation laws of mass and momentum to simulate fluid flow (Frisch *et al*. [11]). Unlike conventional CFD schemes based on discretization of the Navier–Stokes equation or equivalent formulations, such as the streamfunction–vorticity equations, the LBM is based on microscopic models and finds solutions of a "mesoscopic" kinetic equation. Chen and Doolen [12] discuss it in detail. The general form of the lattice Boltzmann equation (LBE) is

$$N_i(\mathbf{x} + \mathbf{c}_i, t + 1) = N_i(\mathbf{x}, t) + \Omega_i(N(\mathbf{x}, t)), \tag{1}$$

where $\Omega_i(N(\mathbf{x}, t))$ is the collision operator and $N_i(\mathbf{x}, t)$ is a single-particle distribution function, which can be considered as the probability of finding a particle with velocity $\mathbf{c}_i$ on lattice site $\mathbf{x}$ at time t. The collision operator has to obey mass and momentum conservation,

$$\sum_i \Omega_i(N(\mathbf{x}, t)) = 0, \tag{2}$$

$$\sum_i \mathbf{c}_i \Omega_i(N(\mathbf{x}, t)) = \mathbf{F}(\mathbf{x}, t), \tag{3}$$

where $\mathbf{F}(\mathbf{x}, t)$ is the external force. The mass density ρ and fluid velocity $\mathbf{u}$ are related to N_i and $\mathbf{c}_i$ as follows:

$$\rho(\mathbf{x}, t) = \sum_i N_i(\mathbf{x}, t), \tag{4}$$

$$\rho(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) = \sum_i \mathbf{c}_i N_i(\mathbf{x}, t). \tag{5}$$

The LBM algorithm involves two main computational steps: the collision step and the propagation step. In the collision step, the particle distribution functions are changed by the collision. The postcollision particle distribution functions are calculated by

$$N_i^c(\mathbf{x}, t) = N_i(\mathbf{x}, t) + \Omega_i(N(\mathbf{x}, t)), \tag{6}$$

where N_i^c denotes the postcollision value of the particle distribution function. In the propagation step, the particles propagate to their neighbor sites. Thus, the distribution functions on a node are updated by their upstream nodes as follows:

$$N_i(\mathbf{x} + \mathbf{c}_i, t + 1) = N_i^c(\mathbf{x}, t). \tag{7}$$

It may be seen that Eqs. (6) and (7) imply Eq. (1).

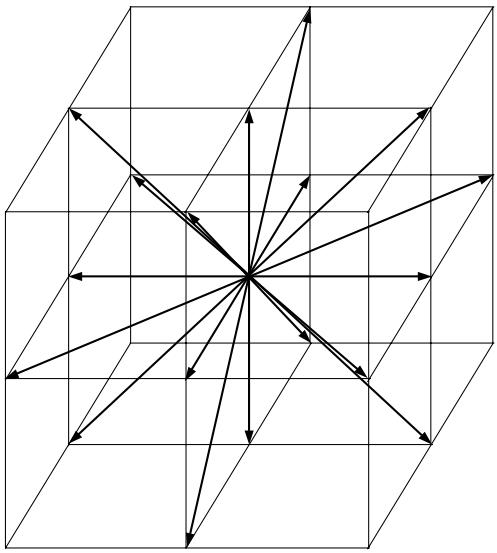


FIG. 1. Three-dimensional FCHC lattice with 18 directions.

The Somers–Eggels formulation of the LBM model is based on the 4D face-centered-hyper-cubic (FCHC) lattice introduced by d'Humieres *et al*. [13]. The velocities of the 4D FCHC lattice can be written

$$\mathbf{c}_i = \text{perm}(\pm 1, \pm 1, 0, 0), \tag{8}$$

where "perm" means the coordinate permutation. The 18 velocities of the three-dimensional projection of the FCHC lattice are given by (see Fig. 1)

$$\mathbf{c}_i = \text{perm}(\pm 1, 0, 0),\ \text{perm}(\pm 1, \pm 1, 0). \tag{9}$$

The differential form of Eq. (1) can be derived from the first-order Taylor expansions of $N_i(\mathbf{x} + \mathbf{c}_i, t + 1)$,

$$\frac{\partial N_i}{\partial t} + c_{i\alpha} \frac{\partial N_i}{\partial \mathbf{x}_\alpha} = \Omega_i(N), \tag{10}$$

where $c_{i\alpha}$ is the component of the ith velocity vector $\mathbf{c}_i$ in the Cartesian coordinate direction α. Repeated indices are understood to be summed over in Eq. (10) and subsequent equations except where noted. The index i in Eq. (10) is not summed over.

Assuming that the magnitude of the fluid velocity, $\mathbf{u}$, is much smaller than unity and the lattice gas is close to equilibrium, they obtained an expression for the collision operator Ω_i in Eq. (1) by using an asymptotic expression for N_i. The asymptotic result for N_i was obtained by Frisch *et al*. [11] using a multiple time-scale analysis together with an expansion of N_i around the equilibrium solution. The results are as follows:

$$N_i = \frac{m_i \rho}{24}\left[1 + 2c_{i\alpha}u_\alpha + 3c_{i\alpha}c_{i\beta}u_\alpha u_\beta - \frac{3}{2}u_\alpha u_\alpha - 6\nu\left(c_{i\alpha}\frac{\partial c_{i\beta}u_\beta}{\partial x_\alpha} - \frac{1}{2}\frac{\partial u_\alpha}{\partial x_\alpha}\right)\right], \tag{11}$$

$$\Omega_i(N) = \frac{m_i \rho}{12}\left(c_{i\alpha}\frac{\partial c_{i\beta}u_\beta}{\partial x_\alpha} - \frac{1}{2}\frac{\partial u_\alpha}{\partial x_\alpha}\right) + \frac{m_i}{12}c_{i\alpha}F_\alpha. \tag{12}$$

A summation over all directions i of Eq. (10) with the mass conservation constraint on Ω_i given in Eq. (2) yields the continuity equation:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_\alpha}{\partial x_\alpha} = 0. \tag{13}$$

From Eqs. (11) and (10) and from the symmetry relations (see Eggels and Somers [14]) with a summation over all velocity directions, one can obtain the following equation:

$$\frac{\partial \rho u_\alpha}{\partial t} + \frac{\partial \rho u_\alpha u_\beta}{\partial x_\beta} = -\frac{\partial p}{\partial x_\alpha} + \frac{\partial}{\partial x_\beta}\left[\nu\rho\left(\frac{\partial u_\beta}{\partial x_\alpha} + \frac{\partial u_\alpha}{\partial x_\beta}\right)\right] - \frac{1}{2}\frac{\partial}{\partial x_\alpha}\left(\nu\rho\frac{\partial u_\beta}{\partial x_\beta}\right) + F_\alpha. \tag{14}$$

Equation (14) corresponds to the Navier–Stokes equation in the incompressible limit, with the following equation of state for the pressure:

$$p = \frac{1}{2}\rho\left(1 - \frac{1}{2}u_\alpha u_\alpha\right). \tag{15}$$

The explicit dependence of the pressure on the velocity is unphysical. In single relaxation time schemes the dependence can be avoided by adding rest particles. However, in practice, it does not create serious problems as long as $|\mathbf{u}_\alpha| \ll 1$.

To solve the LBE, Eggels and Somers [14] introduced a filter matrix E_{ik} and solution vectors $\alpha_k^{\pm}$ by factorizing the asymptotic expression for N_i. The postcollision distribution function $N_i^c(\mathbf{x}, t)$ is calculated as follows:

$$N_i^c(\mathbf{x}, t) = \frac{m_i}{24}\sum_{k=1}^{n} E_{ik}\alpha_k^+(\mathbf{x}, t). \tag{16}$$

The vector α_k^- is calculated as

$$\alpha_k^-(\mathbf{x}, t) = \sum_{i=1}^{n} E'_{ki} N_i(\mathbf{x}, t), \tag{17}$$

where E'_{ki} satisfies

$$\frac{m_i}{24} E_{ik} E'_{ki} = I. \tag{18}$$

The Somers–Eggels formulation computes the particle distribution function by multiplying the two matrices E and $\alpha^{\pm}$. The details of the method may be found in Eggels and Somers [14].

3. NONUNIFORM GRID TECHNIQUE IN LBM

Figure 2 shows a cross section of a nonuniform grid based on the 18-node FCHC lattice. In the propagation step, particles with velocities in the fifth direction would move as indicated in the figure. On the fine portion of the grid, each particle will move to a neighboring lattice point. However, on the coarse portion of the grid, a particle will move to an interstitial position. It is not necessary to perform interpolation to update the single-particle distribution

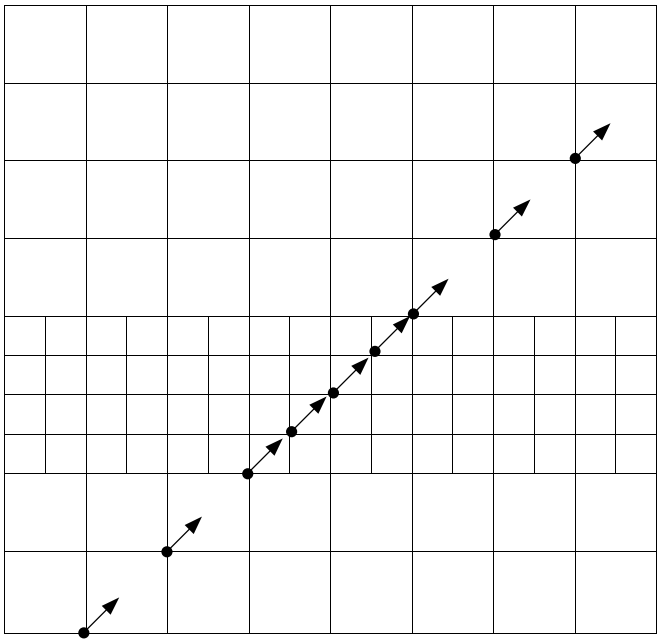


FIG. 2. Particles propagate along a straight line.

function on a fine lattice point unless it is adjacent to a coarse portion of the grid because every node can be updated by its upstream nodes during the propagation step. However, on the coarse portion of the grid, no particle can reach a given lattice point in one time step since the length of the velocity vector is set by the fine grid. Thus, one must use interpolation to update the single-particle distribution function. This approach was developed by He *et al*. [1].

The nonuniform LBM simulations in this paper have several features that were not present in the ISLBE method developed by He *et al*. [1]. The simulations to be reported were performed on a three-dimensional FCHC lattice having 18 directions. This lattice is shown in Fig. 1. One can use one-dimensional interpolation on square or cubic grids if the coarse grid spacings have the same values in all the orthogonal coordinate directions. This is because the particles move along a straight line connecting the nodes in this case (see Fig. 2). Applying quadratic one-dimensional interpolation is the most efficient method for the three-dimensional nonuniform LBM scheme. A second difference is that the ISLBE scheme developed by He *et al*. applies interpolation after the propagation step to redistribute the particle distribution functions, while the new scheme replaces the propagation step with interpolation. In this paper, it will be shown that in addition to reducing the memory requirements by roughly 65%, the total CPU time can be reduced by roughly 75% in the large eddy simulation of a stirred tank.

The quadratic one-dimensional interpolation through three known points $f(x_k)$ $[k = 1, 2, 3]$ is given by

$$y(x) = f(x_0)\frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + f(x_1)\frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + f(x_2)\frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}, \tag{19}$$

where x is the point of interest.

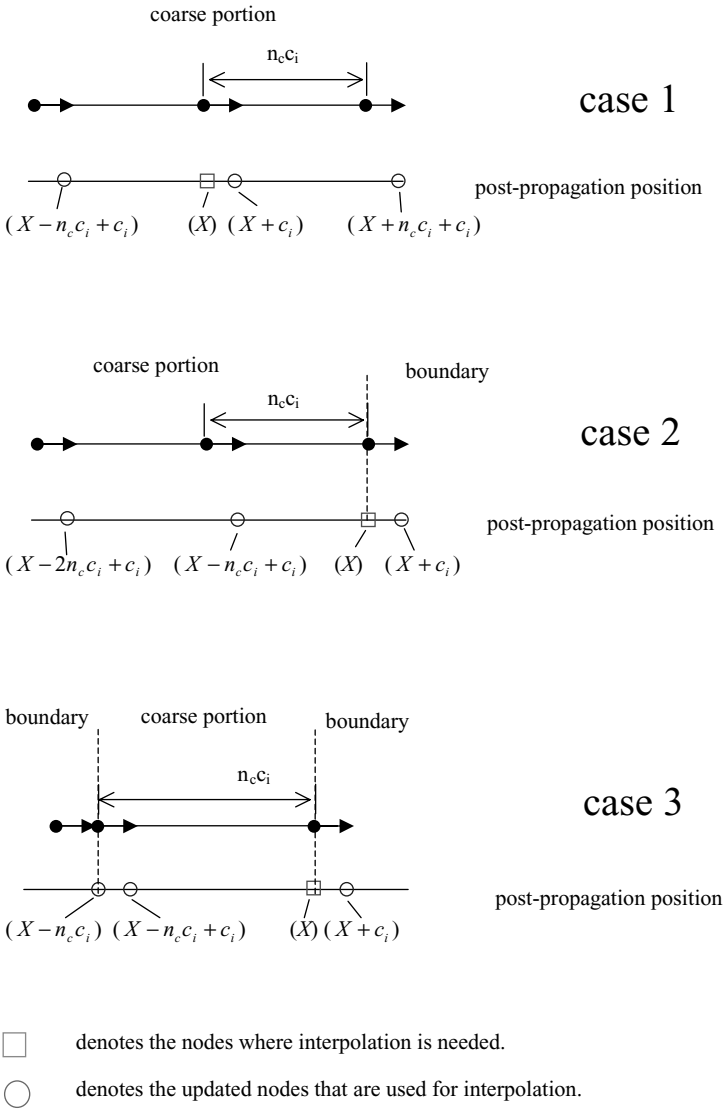


FIG. 3. Three cases in which quadratic one-dimensional interpolation is applied in the nonuniform grid technique.

It will be assumed in what follows that the coarse lattice spacing is an integer multiple of the fine grid spacing. The symbol n_c denotes the ratio of the coarse grid spacing to the fine grid spacing.

One must consider three cases that arise in the implementation of interpolation in the nonuniform grid technique. These cases are shown in Fig. 3. In the coarse portion of the grid, quadratic one-dimensional interpolation can be applied to the three-dimensional cubic grid in our simulation because particles always propagate along a line parallel to a particular velocity. As shown in case 1 of Fig. 3, one can apply Eq. (19) to interpolate the postpropagation distribution function at the coarse grid node that is denoted by the square symbol because no particle can stream there in the propagation step. If **x** represents the three-dimensional position vector of that node, then the position vectors of the three points

that are denoted by circles are $\mathbf{x} - n_c\mathbf{c}_i + \mathbf{c}_i$, $\mathbf{x} + \mathbf{c}_i$, and $\mathbf{x} + n_c\mathbf{c}_i + \mathbf{c}_i$. The values of the postpropagation distribution function of the positions that are denoted by circles are already known after the propagation step. Therefore, one can interpolate the unknown value of the postpropagation function on the position denoted by the square from them. One may replace $y(x)$, $f(x_0)$, $f(x_1)$, $f(x_2)$ by the corresponding postpropagation distribution functions and rewrite Eq. (19) as

$$N_i(\mathbf{x}, t+1) = a_1 N_i(\mathbf{x} - n_c\mathbf{c}_i + \mathbf{c}_i, t+1) + b_1 N_i(\mathbf{x} + \mathbf{c}_i, t+1) + c_1 N_i(\mathbf{x} + n_c\mathbf{c}_i + \mathbf{c}_i, t+1), \tag{20}$$

where

$$a_1 = \frac{n_c + 1}{2n_c^2}, \quad b_1 = \frac{n_c^2 - 1}{n_c}, \quad c_1 = \frac{1 - n_c}{2n_c^2}.$$

Then, one can apply Eq. (7) to Eq. (20) and obtain an expression involving postcollision distribution functions:

$$N_i(\mathbf{x}, t+1) = a_1 N_i^c(\mathbf{x} - n_c\mathbf{c}_i, t) + b_1 N_i^c(\mathbf{x}, t) + c_1 N_i^c(\mathbf{x} + n_c\mathbf{c}_i, t). \tag{21}$$

Case 2 of Fig. 3 shows a situation in which a particle crosses the boundary separating the coarse and fine grids and moves out of the coarse portion. In this case, applying the method used in case 1, one can rewrite Eq. (19) to update the postpropagation distribution function on the position denoted by the square; i.e.,

$$N_i(\mathbf{x}, t+1) = a_2 N_i^c(\mathbf{x} - 2n_c\mathbf{c}_i, t) + b_2 N_i^c(\mathbf{x} - n_c\mathbf{c}_i, t) + c_2 N_i^c(\mathbf{x}, t), \tag{22}$$

where

$$a_2 = -\frac{n_c - 1}{2n_c^2}, \quad b_2 = \frac{2n_c^2 - 1}{n_c^2}, \quad c_2 = \frac{2n_c^2 - 3n_c + 1}{n_c^2}.$$

For some nodes in a corner of the coarse portion, the situation will be like that shown in case 3 of Fig. 3. Applying the same method, one can rewrite Eq. (19) to update the postpropagation distribution function on the square's position; i.e.,

$$N_i(\mathbf{x}, t+1) = a_3 N_i^c(\mathbf{x} - n_c\mathbf{c}_i - \mathbf{c}_i, t) + N_i^c(\mathbf{x} - n_c\mathbf{c}_i, t) - a_3 N_i^c(\mathbf{x}, t), \tag{23}$$

where

$$a_3 = \frac{1 - n_c}{n_c + 1}.$$

Since the right hand sides of Eqs. (21) and (22) involve the postcollision distribution functions N_i^c, one can replace the propagation step with an interpolation step by applying these equations after the collision step.

It is necessary to discuss the fine interface that connects the coarse grid and the fine grid (see Fig. 4). It will be understood in what follows that the interfaces belong to the fine grid. As shown in Fig. 4, the nodes on the fine interface can be updated by their upstream nodes

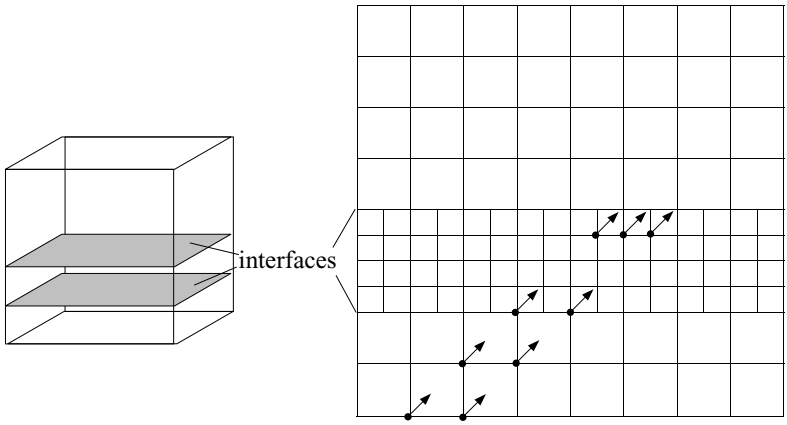


FIG. 4. Particles moving through the interfaces.

in the fine region, but they cannot be updated directly by their upstream nodes in the coarse region. Thus, on the nodes of the fine interfaces, interpolation will be needed to update the distribution functions with velocities that point to the fine region. Since this is the same as case 2, which was discussed previously, one can apply Eq. (22) to this situation. However, as may be seen from Fig. 4, not all of the nodes on a fine interface can be updated by this interpolation method. In such cases, the other fine-grid nodes on the interfaces will be interpolated from their neighboring nodes, which have been updated by interpolation from the neighboring coarse planes. Figure 5 shows a section of an interface and illustrates

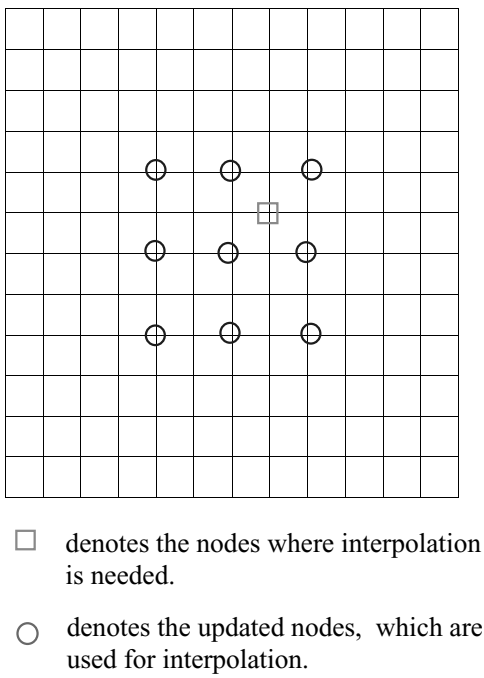


FIG. 5. A section of an interface with the point of interest and nine updated points for further interpolation.

the points that would be used to perform the two-dimensional interpolation that would be needed to update the distribution function.

4. BOUNDARY CONDITIONS

In the LBM, the physical boundary usually lies between lattice points. The neighboring lattice points that are outside the physical boundary constitute the "computational boundary" in this paper. For simplicity, the computational boundary will be referred to as the "boundary" in what follows.

As discussed in Section 2, the LBM has two main computational steps: the collision step and the propagation step. On the boundary nodes, however, no calculation is made for the collision step. The boundary nodes are used only for the implementation of boundary conditions. To implement boundary conditions, another computational step, the boundary condition step, is placed between the two main computational steps. The implementation of boundary conditions is performed in two steps.

1. In the boundary condition step, the particle distribution functions on the boundary are updated from the postcollision distribution functions on the neighboring interior grid nodes. Different boundary conditions require different updating rules.
2. In the subsequent propagation step, the distribution functions at the interior points are set equal to the values of the distribution functions at the upstream boundary points.

Therefore, different boundary conditions have significantly different effects on the interior nodes.

4.1. Boundary Conditions for the LBM

The bounce-back boundary condition imposes a no-slip boundary condition that requires the fluid locally to have the same tangential velocity as the wall. In the boundary condition portion of a time step, the postcollision distribution function at a boundary grid point with a velocity directed toward a neighboring interior point is set equal to the postcollision distribution function with the opposite velocity at the interior point. If the wall is stationary, the bounce-back boundary condition can be enforced on the boundary nodes by

$$N^c_{\xi}(\mathbf{x}_b, t) = N^c_{\xi^-}(\mathbf{x}_{nb}, t), \tag{24}$$

where ξ and ξ^- represent opposite directions, $\mathbf{c}_{\xi}$ are the velocities moving into the computational domain, and $\mathbf{c}_{\xi^-}$ are the velocities moving out (see Fig. 6), N^c are postcollision

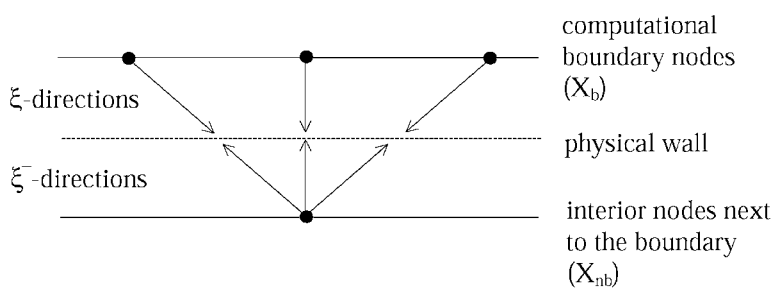


FIG. 6. Bounce-back boundary condition.

particle distribution functions, $\mathbf{x}_b$ is the position of a boundary node, and $\mathbf{x}_{nb}$ is the interior node next to the boundary. It follows from their definitions that

$$\mathbf{x}_{nb} = \mathbf{x}_b + \mathbf{c}_\xi. \tag{25}$$

In the subsequent propagation step, the interior node next to the boundary will be updated by the boundary nodes as follows:

$$N_\xi(\mathbf{x}_{nb}, t+1) = N^c_\xi(\mathbf{x}_b, t). \tag{26}$$

Therefore, one can combine Eqs. (24) and (26) and obtain

$$N_\xi(\mathbf{x}_{nb}, t+1) = N^c_{\xi^-}(\mathbf{x}_{nb}, t). \tag{27}$$

If the wall is moving with velocity $\mathbf{u}_b$, one has to modify Eq. (27) to enforce the moving-wall no-slip boundary condition. In the Somers–Eggels scheme, the equation for the moving-wall no-slip boundary condition can be obtained from Eq. (11) (see Eggels and Somers [14]):

$$N_\xi(\mathbf{x}_{nb}, t+1) = N^c_{\xi^-}(\mathbf{x}_{nb}, t) + \frac{m_\xi \rho(\mathbf{x}_{nb}, t)}{6}[\mathbf{c}_\xi \cdot \mathbf{u}_b]. \tag{28}$$

The symmetric boundary condition is applied on a stress free surface. It is also called the free-slip boundary condition. In the boundary condition step, the postcollision distribution functions at boundary grid points with velocities directed toward a neighboring interior point are set equal to the postcollision distribution function with the symmetric velocity at the interior point. Figure 7 shows the implementation of the symmetric boundary condition. It can be expressed as

$$N^c_\xi(\mathbf{x}', t) = N^c_{\xi^s}(\mathbf{x}, t), \tag{29}$$

where ξ^s is the direction symmetric to ξ and $\mathbf{x}'$ are the boundary positions that are symmetric to $\mathbf{x}$ across the physical boundary.

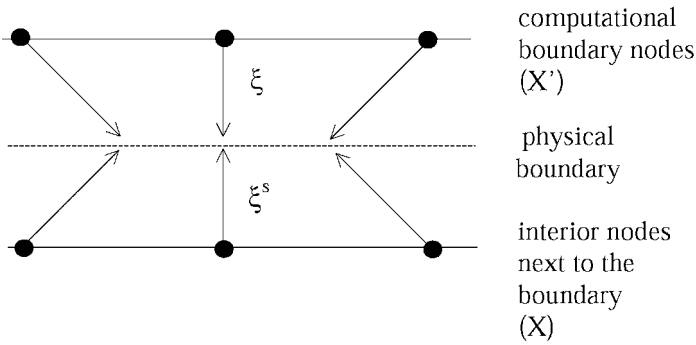


FIG. 7. Symmetric boundary condition.

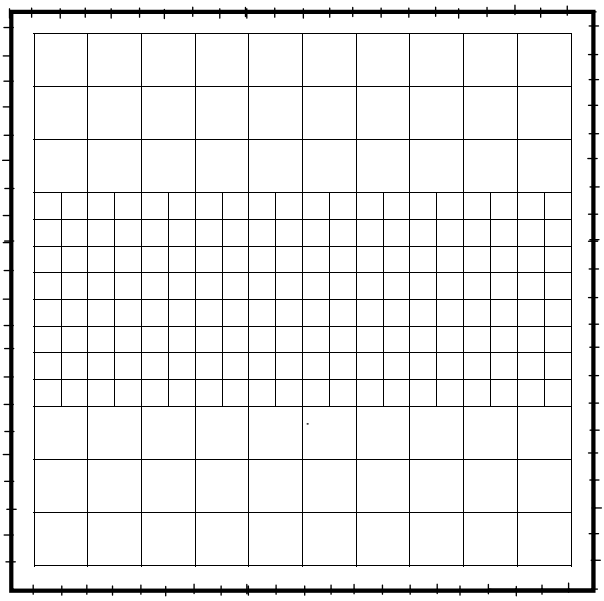


FIG. 8. Structure of the nonuniform grid.

4.2. Boundary Conditions for the Nonuniform Grid Technique

To implement the boundary conditions for coarse grids, it is helpful to discuss the imposition of these boundary conditions in the LBM. In this paper, the boundaries of both the coarse grid and the fine grid are fine and the distance between the boundary and the interior nodes next to the boundary is equal to a fine lattice spacing. This is shown in Fig. 8. As discussed previously in this section, the boundary conditions are performed in two steps. For the fine grid, no matter what updating rules are applied in the boundary condition step, the updated particles on the boundary nodes can always stream to the interior fine grid nodes. However, for the coarse grid, only some of the boundary nodes can be updated in the boundary condition step (see Fig. 9). As a result, for most updating rules, the updated particles on the boundary nodes can only stream to the interstitial positions of the coarse grid next to the boundary in the subsequent propagation step (see Fig. 10).

For the no-slip boundary condition, Eq. (27) shows that the postpropagation distribution at an interior node next to the boundary can be updated by the postcollision distribution function at the same position. The values of $N^c_{\xi-}(x_{nb}, t)$ are known for the coarse grid. Thus, interpolation is not needed in this case.

However, for the free-slip boundary condition, let us first consider the boundary condition step. Figure 9 shows that, when applying the free-slip boundary condition, the distribution

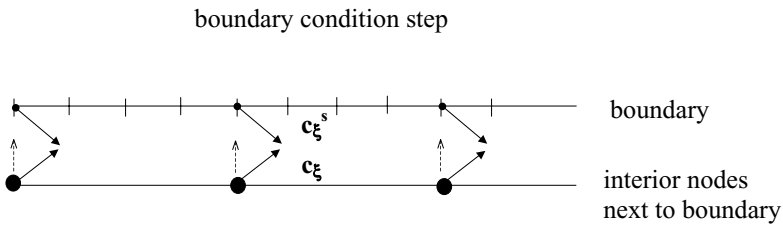


FIG. 9. Implementation of free-slip boundary condition for the coarse grid in the boundary condition step.

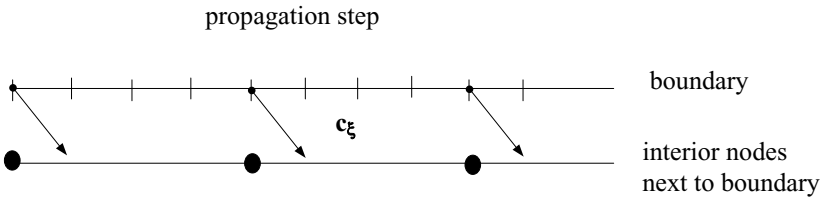


FIG. 10. Implementation of free-slip boundary condition for the coarse grid in the propagation step.

function for velocity $\mathbf{c}_\xi$ on a boundary node will be set equal to the distribution function for velocity $\mathbf{c}_{\xi^s}$ on the neighboring interior node. In the propagation step, the updated boundary nodes update interstitial sites, as shown in Fig. 10. Thus, one needs to apply interpolation to update the coarse grid nodes.

- One may apply interpolation to the nodes that are denoted by squares on the neighboring coarse grid before the boundary condition step, as shown in Fig. 11.
- In the boundary condition step, the value of the distribution function at the interpolated coarse nodes will be assigned to the corresponding nodes on the boundary. (This is shown by the arrow ↑ in Fig. 11.)
- The boundary nodes will update coarse grid nodes in the propagation step.

5. PARALLEL COMPUTATION FOR THE NONUNIFORM GRID

To perform the parallelization, one needs to send border values of a process to neighboring processes and receive border values from neighboring processes in each time step because the computation in a process may need the border values of its neighboring processes. According to the definition of ghost points, "the elements of the array that are used to hold data from other processes" (see Gropp *et al.* [15]), one may refer to the border values of the neighboring processes as ghost points. Thus, in performing parallel computation, one may

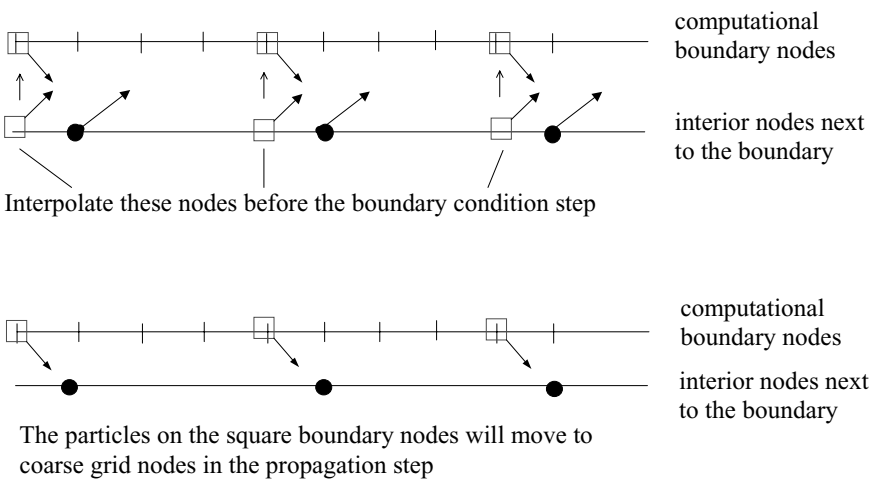


FIG. 11. Applying interpolation for the free-slip boundary condition in the coarse portion.

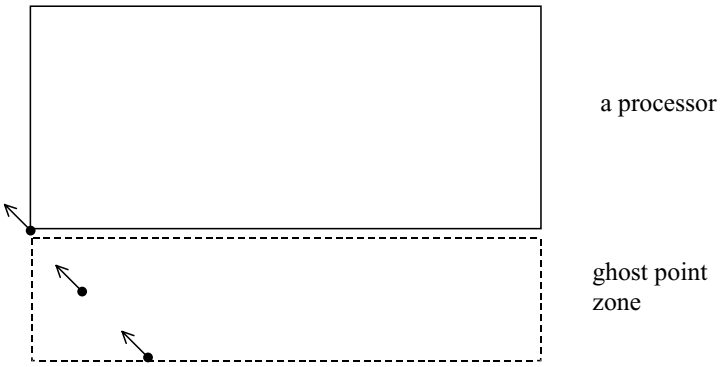


FIG. 12. Particles propagate between two processes.

need to transfer the data of some ghost point lines or areas, or ghost point zones in some three-dimensional cases.

In the uniform grid LES program described by Derksen and Van den Akker [4], the data transfer of some ghost point planes is applied in the message-passing step. However, in a three-dimensional parallel implementation in the coarse portion of the grid, one should transfer the data of some coarse ghost point zones instead of some ghost point planes. This is because, in the worst case, the interpolation for a border coarse grid node needs two coarse ghost points (see Fig. 12).

However, the cost of the data transfer operation is still small because the number of nodes in the coarse ghost point zone that contains two planes of the coarse grid is still less than that of a fine ghost point plane if $n_c > \sqrt{2}$. Figure 13 shows a two-step process to transfer data. Dashed boxes indicate the ghost point zones; the data to be moved is shaded (see Gropp *et al*. [15]).

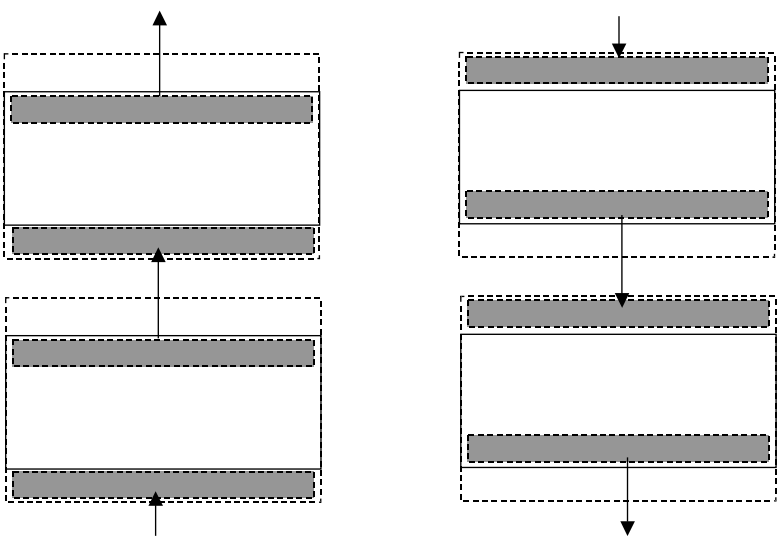


FIG. 13. Two-step process to transfer data.

6. ARBITRARY COMPUTATIONAL DOMAIN METHOD

In many cases, the computational domain is a complicated geometry. If one uses a rectangular or cubic computational domain for the calculation in these cases, a large portion of the computational cost would be associated with points outside the region of physical interests.

Based on earlier work by Ladd [16], an arbitrary computational domain method was developed to solve this problem. The method is as follows:

1. Determine the functions that describe the boundaries of the computational domain.
2. Classify the lattice points that are inside the actual computational domain and outside the domain. (For example, one may introduce a symbol $\beta(\mathbf{x})$ to perform the classification by assigning the values of $\beta(\mathbf{x}) = 1$ for the region of physical interest and $\beta(\mathbf{x}) = 0$ for the points outside the region of physical interest.)
3. For some velocities, the particles on the interior nodes next to the boundary will move out of the interior region of physical interest. Test each velocity and record the information of the positions of these nodes, $\mathbf{x}_{nb}$, the velocities $\mathbf{c}_{\xi^-} = -\mathbf{c}_\xi$ for which the particles on these nodes could move out of the actual computational domain, and the positions when they move out of the interior region of physical interest $\mathbf{x}_{nb} + \mathbf{c}_{\xi^-}$, where the subscript ξ^- denotes the direction pointing from the neighboring interior nodes to the boundary. The subscript nb indicates the position of the interior nodes next to the boundary.

All the operations above are performed in a separate program with the purpose of providing the necessary information for the main computational program to apply boundary condition and skip the lattice nodes outside the region of physical interest. Therefore, there is no additional cost to the main computational program. To illustrate how the main program can use the information above to apply boundary conditions, consider the no-slip boundary condition as an example. The key to applying boundary conditions is to enable the particles on the boundary nodes to stream to the neighbor interior nodes in the propagation step. It could be written as

$$N_\xi(\mathbf{x}_{nb}, t+1) = N^c_\xi(\mathbf{x}_{nb} - \mathbf{c}_\xi, t), \tag{30}$$

where the subscript ξ denotes the direction pointing from the boundary to the neighboring interior nodes, where ξ^- always indicates the direction opposite to ξ, and where $\mathbf{x}_{nb} - \mathbf{c}$ denotes the corresponding upstream boundary nodes and is equal to $\mathbf{x}_b$ (see Fig. 14).

To apply Eq. (30), one should obtain the value of $N^c_\xi(\mathbf{x}_{nb} - \mathbf{c}_\xi, t)$ before the propagation step. For the no-slip boundary condition, one may calculate $N^c_\xi(\mathbf{x}_{nb} - \mathbf{c}_\xi, t)$ in the previous boundary condition step as follows:

$$N^c_\xi(\mathbf{x}_{nb} - \mathbf{c}_\xi, t) = N^c_{\xi^-}(\mathbf{x}_{nb}, t). \tag{31}$$

The quantities of $\mathbf{x}_{nb} - \mathbf{c}_\xi$, $\mathbf{x}_{nb}$, and ξ^- are known from step 3. Thus, the main computational program can use the information to apply the no-slip boundary condition to complicated geometries correctly and skip the lattice nodes outside the region of physical interests by only calculating the nodes with $\beta(\mathbf{x}) = 1$.

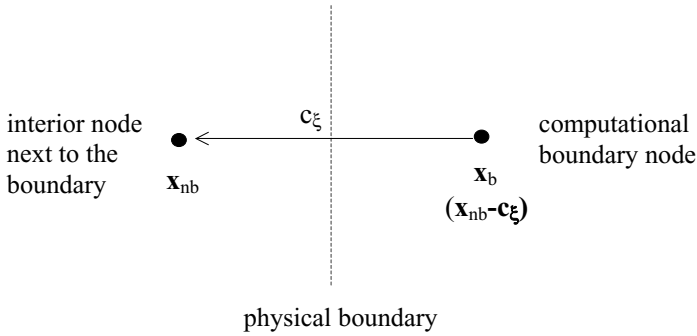


FIG. 14. The particles on the boundary nodes stream to the neighbor interior coarse nodes in the propagation step.

7. SIMULATIONS

The flow simulation was performed with a modification of a program developed by Derksen and Van den Akker [4] to simulate a stirred tank. The program used by Derksen and Van den Akker makes use of a version of the LBM described by Eggels and Somers [14]. The program also makes use of the Smagorinsky [17] subgrid scale model to account for turbulent eddies with length scales smaller than the computational lattice spacing. Eggels [3] computed statistical results for the mean flow. Derksen and Van den Akker [4] computed time-averaged and phase-averaged results for the mean flow and turbulence statistics. These quantities have been computed in the present study to permit a comparison with the above results.

7.1. Computational Procedure

The LES program consists of six main computational steps in each time step.

1. Determine the elements of the solution vector $\alpha_k^-(\mathbf{x}, t)$ from the postpropagation distribution function $N_i(\mathbf{x}, t)$ of the last time step using Eq. (17).
2. Compute the elements of the solution vectors $\alpha_k^+(\mathbf{x}, t)$ from $\alpha_k^-(\mathbf{x}, t)$ (see Somers [18] and Eggels and Somers [14]).
3. Compute the postcollision distribution function $N_i^c(\mathbf{x}, t)$ field from $\alpha_k^+(\mathbf{x}, t)$ using Eq. (16).
4. Implement boundary conditions to update $N_i^c(\mathbf{x}, t)$ on the boundary nodes.
5. In the message-passing step, send border values of $N_i^c(\mathbf{x}, t)$ to neighbors and receive border values from neighbors in order to implement parallel computation.
6. In the propagation step, the postcollision distribution function $N_i^c(\mathbf{x}, t)$ associated with the grid point at position $\mathbf{x}$ moves to the grid point at $\mathbf{x} + \mathbf{c}_i$ to become $N_i(\mathbf{x} + \mathbf{c}_i, t + 1)$ for the next time step.

Most modifications have been made on the last three steps to perform a nonuniform LES. In the boundary condition step, the bounce-back boundary condition is imposed on the tank wall, baffle plates, and bottom of the tank, the free-slip boundary condition is applied on the top of the tank, and the adaptive force-field method (Derksen and Van den Akker [4]) is implemented on the impeller. The details of the implementation of boundary conditions for the nonuniform grid are discussed in Sections 4.1 and 4.2. The interpolation methods

discussed in Section 3 are applied in the propagation step to update the distribution functions in the coarse portion of the grid. The arbitrary computational domain method discussed in Section 6 is applied to set up the geometry of the tank wall and the baffles and to eliminate the computation of the nodes outside the tank. In the message-passing step, the method described in Section 5 is implemented to handle the message passing in the coarse portion.

7.2. Large Eddy Simulation

A direct simulation of a stirred tank with an industrially relevant Reynolds number ($\mathrm{Re} > 10^4$) is not feasible because of the enormous numbers of grid points and time steps that would be required. However, in a large eddy simulation (LES), the effect that the small scales have on the flow is assumed to be universal and isotropic so that one may use a simple subgrid-scale model. Thus, the large eddy simulation can be performed with a relatively smaller resolution and fewer time steps. In this research, the standard Smagorinsky eddy viscosity model (Smagorinsky [17]) is applied. The model uses a subgrid-scale eddy viscosity ν_t,

$$\nu_t = \lambda_{\mathrm{mix}}^2 \sqrt{S^2}, \tag{32}$$

where λ_{mix} is the mixing length of subgrid-scale motion and S^2 the resolved deformation rate,

$$S^2 = \frac{1}{2}\left(\frac{\partial u_\alpha}{\partial x_\beta} + \frac{\partial u_\beta}{\partial x_\alpha} - \frac{2}{3}\delta_{\alpha\beta}\nabla \cdot \mathbf{u}\right)^2, \tag{33}$$

where $\delta_{\alpha\beta}$ is the Kronecker delta. The ratio between the mixing length and the lattice spacing Δ is set to a constant:

$$c_s = \frac{\lambda_{\mathrm{mix}}}{\Delta}. \tag{34}$$

In the present study, c_s was chosen to be 0.12, which is within the range of commonly used values in shear-driven turbulence (see Piomelli *et al.* [19]). This value was selected to facilitate comparisons with the results obtained by Derksen and Van den Akker. In the coarse portion of the nonuniform grid, the values of the lattice spacing and the mixing length are twice as big as those in the fine portion. The total viscosity is $\nu + \nu_t$ in the subgrid-scale model. Through the solution vectors of the Somers–Eggels LBM scheme, the stresses that are needed for the calculation of ν_t can be obtained locally. Thus, the large eddy simulation does not destroy the locality of the lattice Boltzmann scheme.

7.3. Parameter Choices

The configuration of the stirred tank together with a sketch of the front and top view of the impeller is shown in Fig. 15. The Reynolds number is defined as $\mathrm{Re} = ND^2/\nu$, where N is the rotational speed of the impeller, D is the impeller diameter, and ν is the kinematic viscosity of the flow. The four baffles are placed at the perimeter of the tank. The axial level $z = 0$ corresponds to the impeller disk plane.

Table I gives the values of the geometrical quantities, namely the tank height and diameter, the impeller diameter, the blade dimensions, the width of the tank baffles, and the diameter of the impeller shaft. In all cases, the quantities are made dimensionless in terms of the

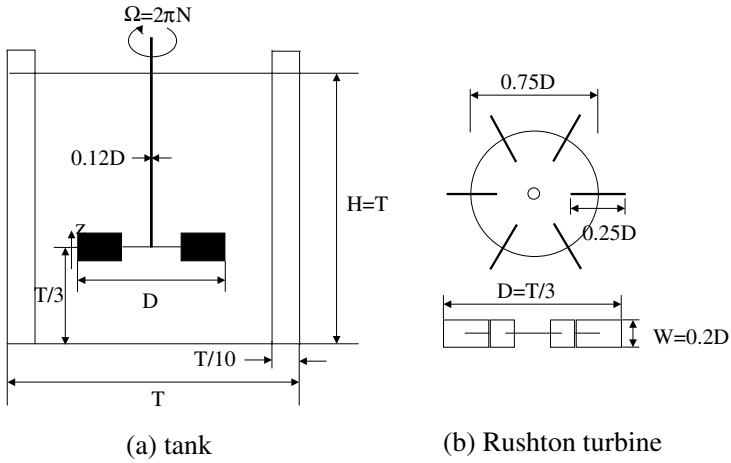


FIG. 15. The geometry of the tank and the Rushton turbine.

lattice spacing. The values of the quantities in the simulations reported by Derksen and Van den Akker and by Eggels are also given in Table I.

The rotational speed, N, of the impeller is limited by the incompressibility condition in the LBM $|\mathbf{u}|^2 \ll 1$. (In the lattice units used in this paper, the speed of sound is unity.) The impeller tip speed, $v_{\text{tip}} = \pi ND$, should be much smaller than unity. The Reynolds number can be varied by varying the kinematic viscosity. In this study, the Reynolds number was 29,000 because of the availability of experimental data (Wu and Patterson [2] and Derksen *et al*. [9, 10]) and simulation results (Derksen and Van den Akker [4]). Table II gives the values of the dimensionless physical parameters, namely the kinematic viscosity of the fluid, the density of the fluid, the impeller cyclic velocity, the impeller tip velocity, and the Reynolds number. The corresponding values for the Derksen–Van den Akker simulation and Eggels' simulation are given for comparison.

In Derksen *et al*.'s and Eggels' studies, low-resolution simulations were started from a quiescent state and continued until steady state was achieved. Then, the flow fields were interpolated to a higher resolution grid. Finally, high-resolution simulations were started from the interpolated flow fields. However, for the nonuniform simulation, since the coarse mesh covers most of the computational domain, one can perform the simulation directly from a quiescent state. According to the temporal monitoring of the velocity and the

TABLE I
Parameters for the Geometry of the Stirred Tank

	Nonuniform grid simulation	Derksen *et al*.'s simulation	Eggels' simulation
Tank height	240	180	240
Tank diameter	240	180	240
Impeller diameter	80	60	80
Length of blades	20	15	20
Width of blades	16	12	16
Width of tank baffles	24	18	24
Diameter of the impeller shaft	9.6	7.2	0

TABLE II
Parameters for the Physical Properties of the Flow

	Nonuniform grid simulation	Derksen *et al.*'s simulation	Eggels' simulation
Kinematic viscosity	7.76×10^{-5}	7.356×10^{-5}	3.988×10^{-5}
Density	8	8	
Impeller cyclic velocity	$2\pi/3000$	$2\pi/1600$	$2\pi/1500$
Impeller tip velocity	0.084	0.12	0.17
Reynolds number	29000	29000	107000

energy levels in a subsection of a stirred tank (Eggels [3]), at least 30 impeller revolutions are needed to reach a quasi-steady state when starting from quiescent states. Thus, the nonuniform grid LES starts from a quiescent state and runs for 30 revolutions to develop a steady state. In the nonuniform grid simulation, the impeller makes a full revolution in 3000 time steps. Therefore the impeller tip velocity is much smaller than that of Derksen and Van den Akker's and Eggels's simulations. A small tip velocity will be beneficial for multiphase stirred tank simulations that will be performed in the future because large velocity fields will create numerical instability for multiphase simulations with the LBM. The simulation data were collected and statistically processed for 20 revolutions after reaching the steady state. This sampling period is smaller than that of Derksen *et al.*'s and Eggels' simulations only because of the CPU time limits. Table III gives computational parameters for the grids that were considered in the present study, namely the number of time steps for a revolution, the number of revolutions to reach a steady state, the number of revolutions to collect the statistics, the numbers of grid points in the x, y, and z directions, the total number of grid points, and the ratio of the total number of grid points for the nonuniform simulation to those in the other simulations.

TABLE III
Computational Parameters

		Derksen and Van den Akker's simulation		Eggels' simulation	
	Nonuniform grid simulation	Low resolution	High resolution	Low resolution	High resolution
Number of time steps for a revolution	3000	1000	1600	750	1500
Number of revolutions to reach a steady state	30	20	15	30	20
Number of revolutions to collect the statistics	20		25		40
Number of grid points in the axial direction (x-direction)	92 in the coarse portion, 57 in the fine portion	120	180	120	240
Number of grid points in other directions (y,z-direction)	120 in the coarse portion, 240 in the fine portion	120	180	120	240
Total number of grid points	4,608,000	1,728,000	5,832,000	1,728,000	13,824,000
Normalized total number of grid points	2.667	1	3.375	1	8

TABLE IV
Computational Parameters

		Uniform grid simulation	
	Nonuniform grid simulation	180^3 grid	240^3 grid
Total number of grid nodes	4,608,000	5,832,000	13,824,000
Normalized number of grid nodes	1	1.27	3
Normalized CPU time for a time step	1	1.4	3.8
Normalized CPU time for a revolution	1	0.75	1.9

8. RESULTS

In the nonuniform simulation, one-dimensional quadratic interpolation is implemented in the coarse portion of the grid using the method in Section 4. Based on computational tests, the CPU time consumed on the interpolation for a coarse node is almost the same as that of the calculation for a fine node in the propagation step. This accounts for much of the efficiency of the interpolation method. Computational tests have also been made

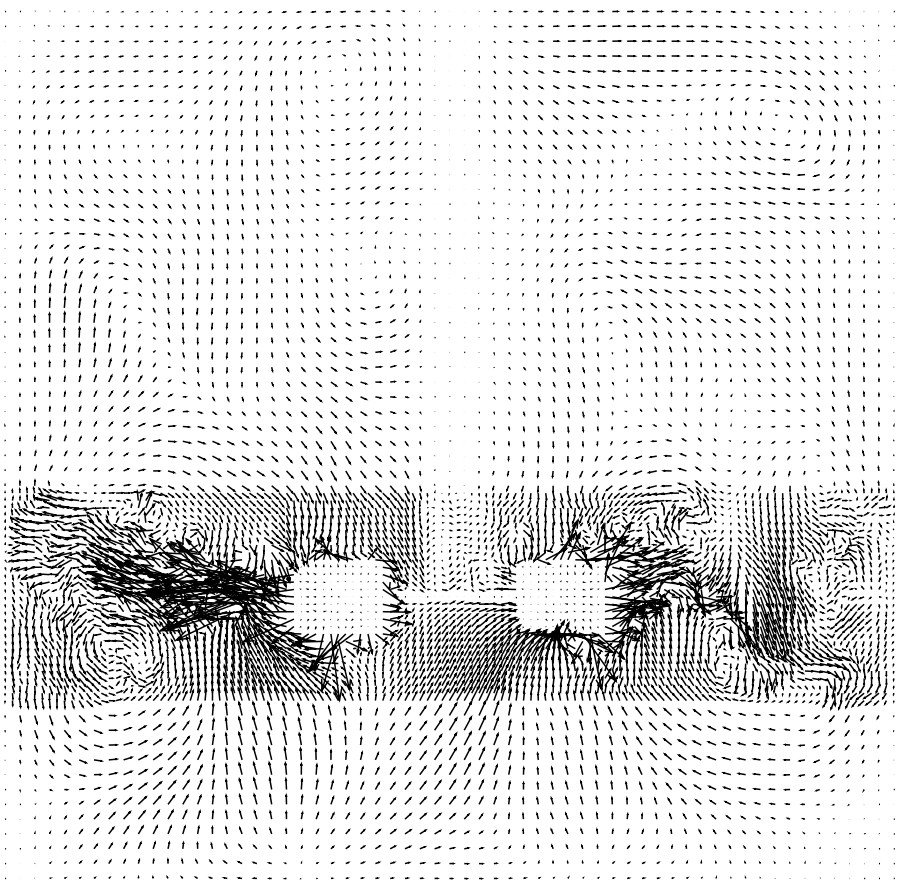


FIG. 16. An instantaneous velocity field in a vertical cross section of the stirred tank.

for uniform simulations with different resolutions to compare their CPU time. The uniform simulations with 180^3 grid points and 240^3 grid points represent the simulations of Derksen and Van den Akker [4] and Eggels [3], respectively. The results are shown in Table IV.

The general flow pattern for the nonuniform simulation of the stirred tank is illustrated in Figs. 16–19, showing instantaneous and phase-averaged velocity fields in a vertical cross section of the tank and in the horizontal plane $z = 0$ where the center of the impeller is located (see Fig. 15). In order to show the flow structure clearly, these plots are shown with only half of the resolution of the nonuniform simulation.

In Fig. 16, strong radial outflows as well as trailing vortices can be observed. The trailing vortices are produced near the edge of the impeller blades and sent out by the strong outflows toward the tank wall. It seems that the vortices can travel over a long distance until they come near the tank wall. One can also identify the shape of the shaft and the impeller. The continuous motion of flow passing through the interfaces of the coarse and fine grid suggests that the method for handling the interaction of the coarse grid and the fine grid is proper. Several trailing vortices can be observed in Fig. 17. Figures 18 and 19 show the mean flow.

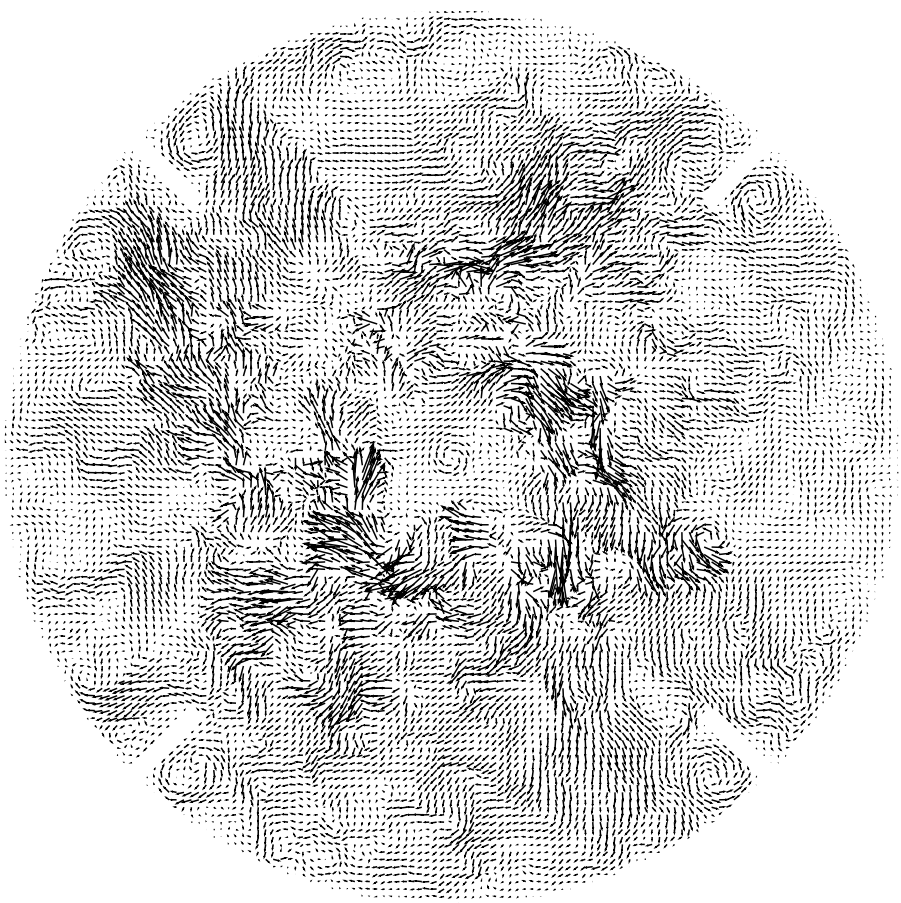


FIG. 17. A velocity field of the turbulent flow in a horizontal plane that contains the middle of the impeller ($z = 0$).

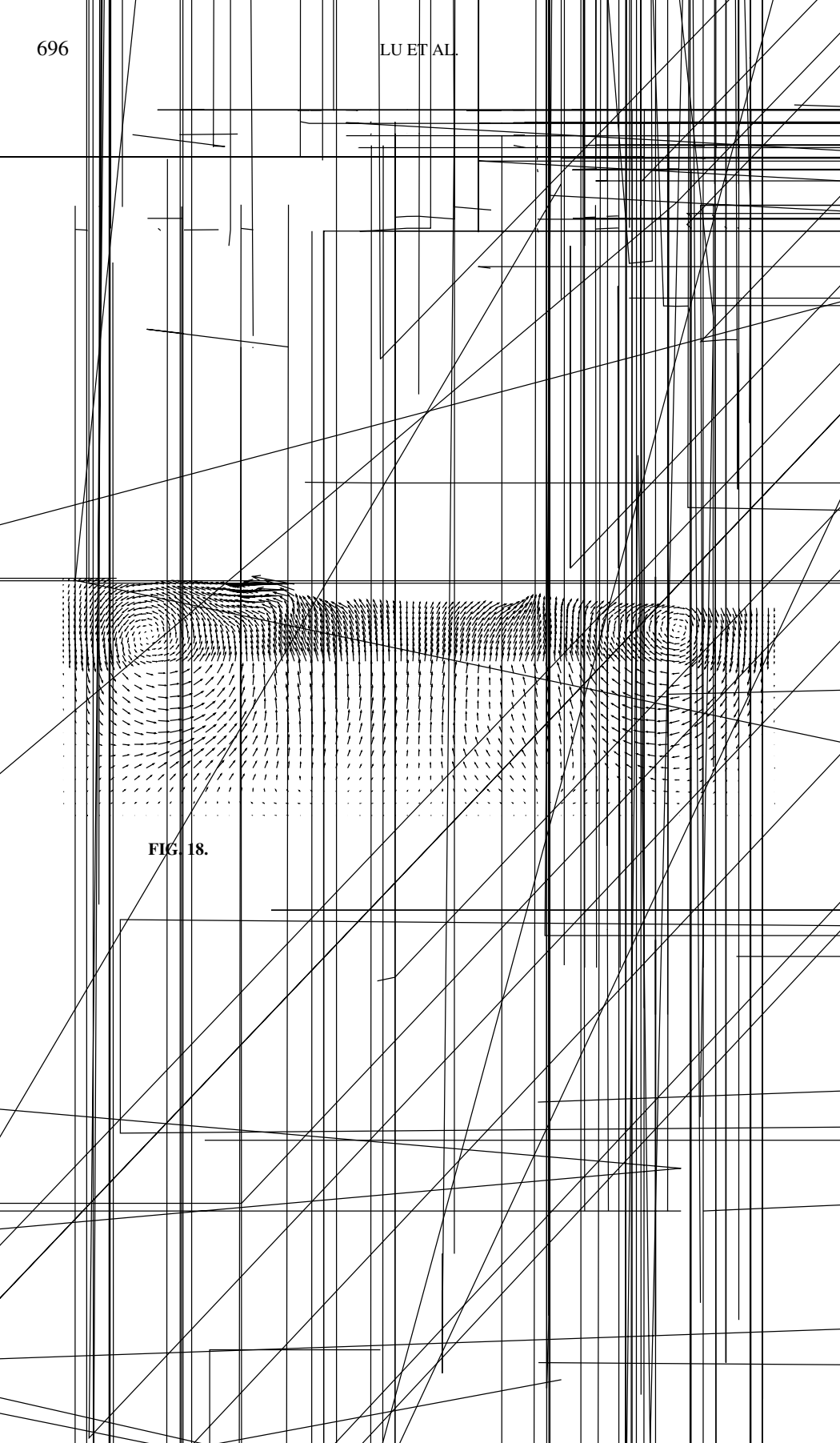
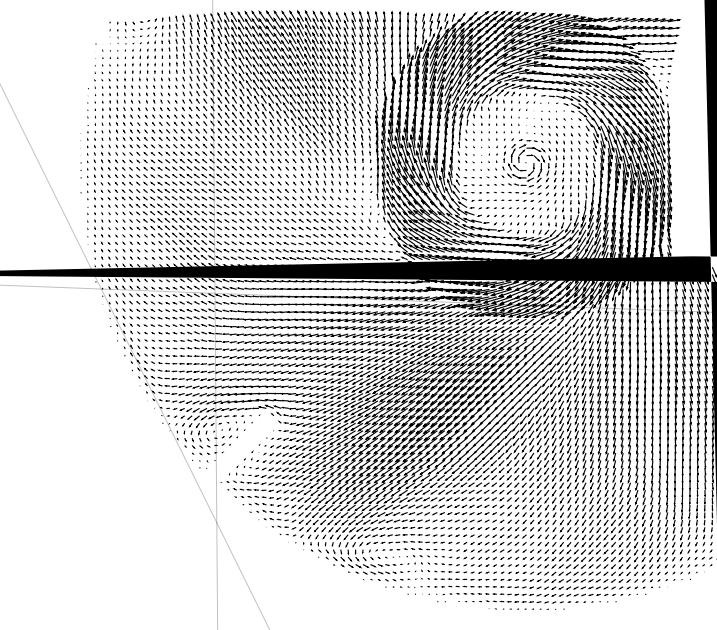
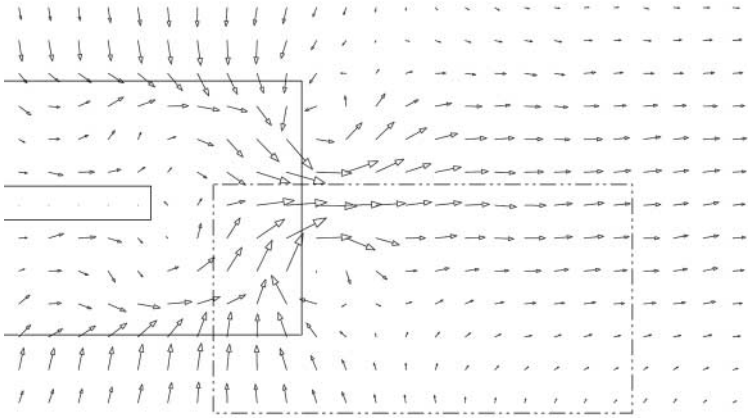
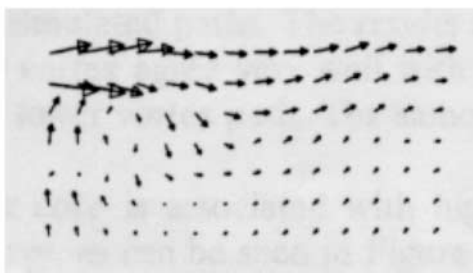


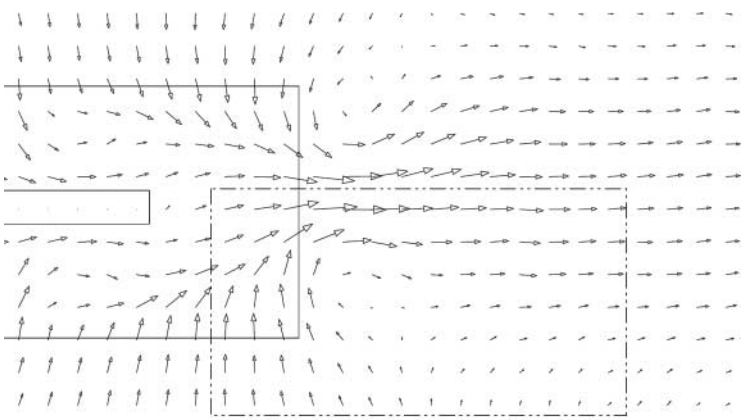
FIG. 18.





$\theta = 10^{\circ}$





$\theta = 19^{\circ}$

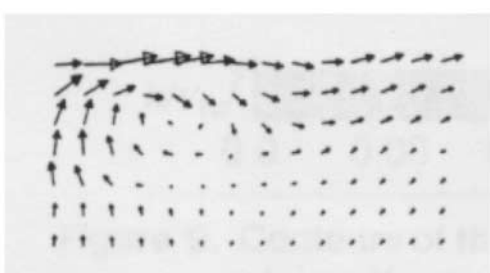
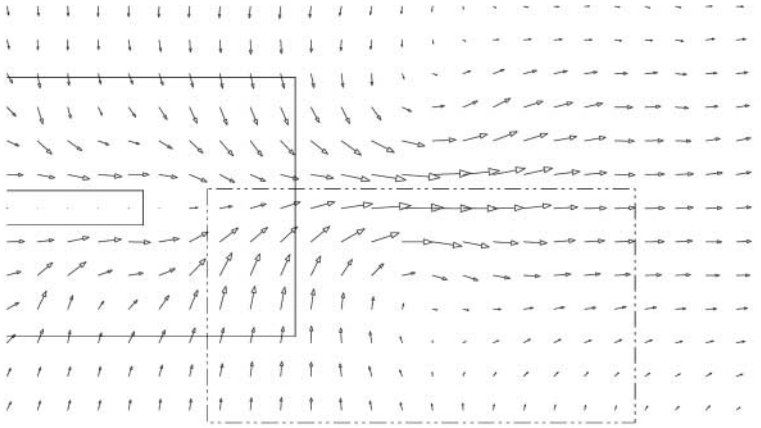
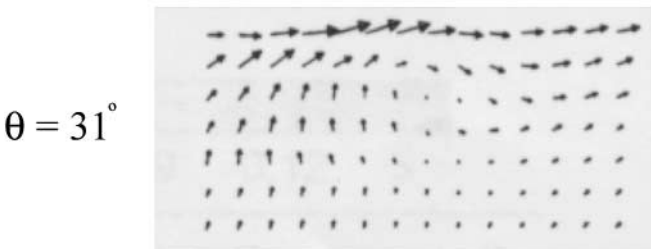
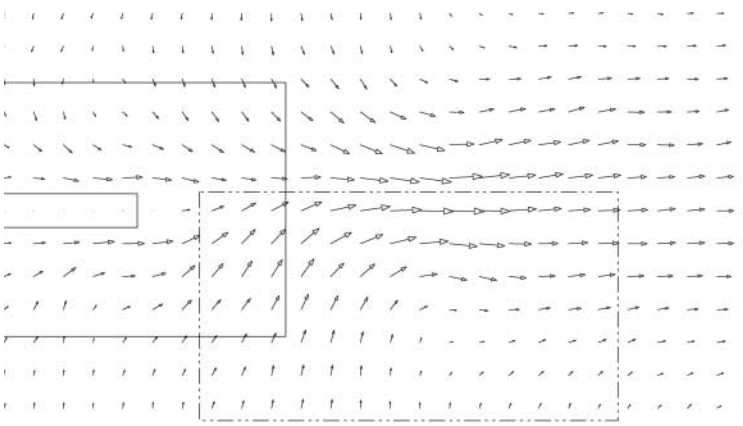


FIG. 20. Phase-resolved flow fields in the vicinity of the impeller for $\theta = 10^{\circ}$, $\theta = 19^{\circ}$, $\theta = 31^{\circ}$, and $\theta = 40^{\circ}$. The flow fields on the top are the nonuniform simulation results. The bottom fields are experimental results by Derksen *et al.* [9, 10].







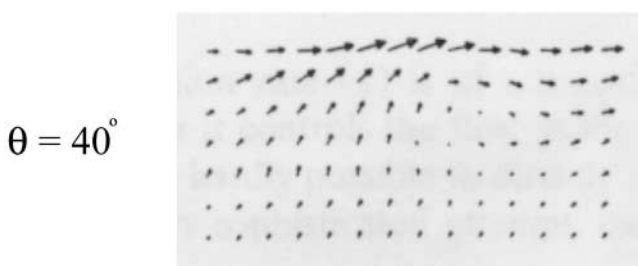


FIG. 20—*Continued*

TABLE V
Deviation for Radial Velocity

		$2z/w$							
$2r/D$	Simulations	−1.6	−1.1	−0.55	0.0	0.55	1.1	1.6	Average
1.07	10 revolutions	8%	13%	12%	2%	3%	3%	10%	7.3%
	20 revolutions	7%	12%	15%	0%	1%	1%	6%	6%
	Eggels' simulation	3%	10%	12%	9%	0%	4%	4%	6%
1.29	10 revolutions	10%	2%	14%	6%	6%	0%	4%	6%
	20 revolutions	11%	2%	18%	2%	3%	0%	1%	5.3%
	Eggels' simulation	1%	2%	8%	8%	8%	0%	1%	4%
1.50	10 revolutions	2%	11%	19%	6%	9%	8%	0%	7.7%
	20 revolutions	3%	0%	11%	0%	19%	9%	0%	6%
	Eggels' simulation	4%	6%	5%	9%	11%	8%	1%	6.2%
1.66	10 revolutions	2%	15%	19%	2%	21%	23%	4%	12.3%
	20 revolutions	2%	2%	6%	4%	27%	23%	6%	10%
	Eggels' simulation	4%	6%	6%	8%	10%	13%	0%	6.7%

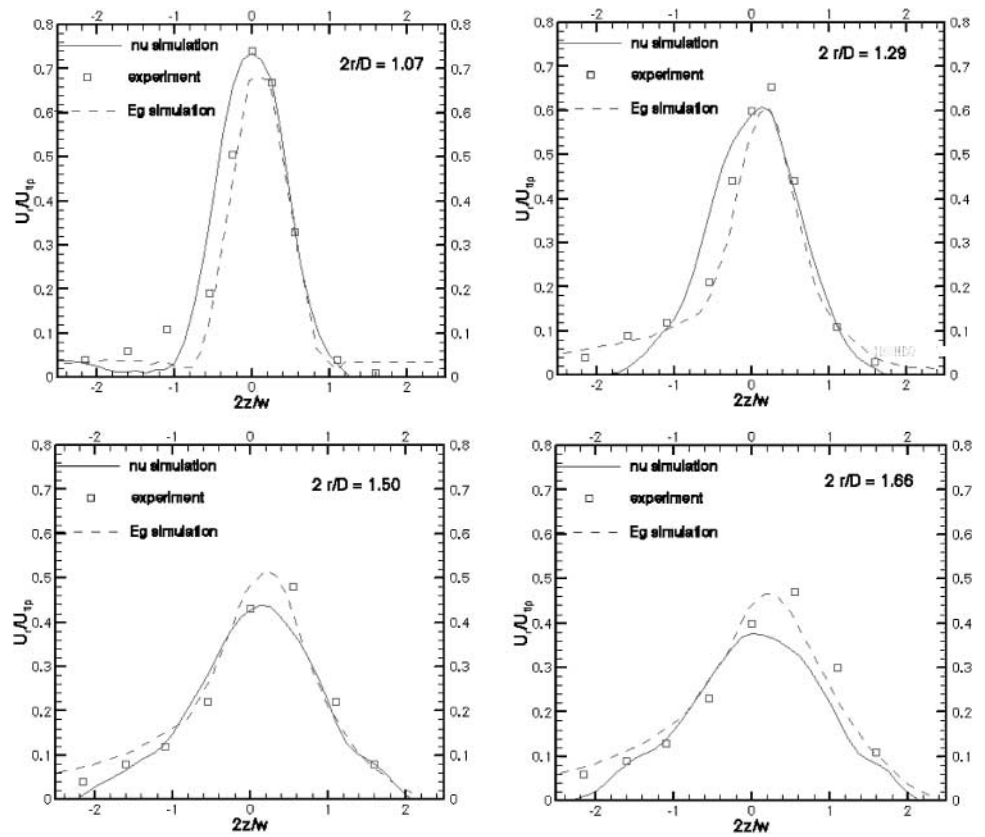


FIG. 21. Comparison between simulations and experimental results for the mean radial velocity profiles in the impeller outstream (nu, nonuiform grid simulation; experiment, Wu and Patterson [2]; Eg, Eggels [3]).

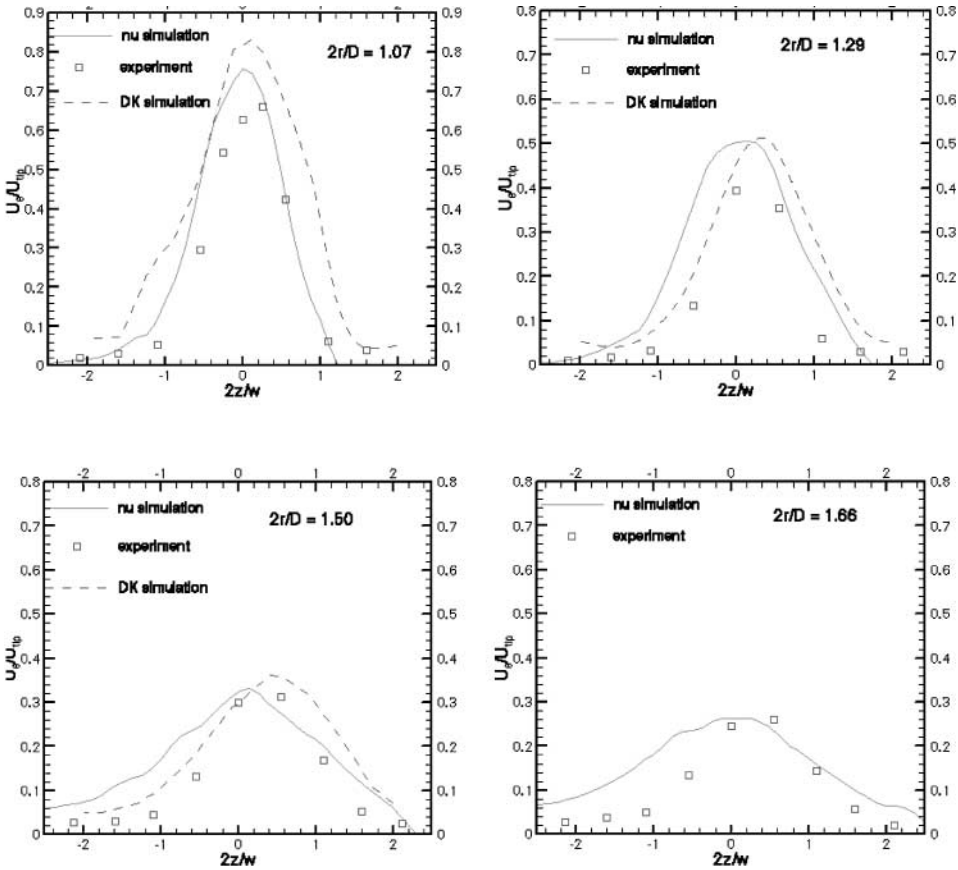


FIG. 22. Comparison between simulations and experimental results for the mean tangential velocity profiles in the impeller outstream (nu, nonuniform grid simulation; experiment, Wu and Patterson [2]; DK, Derksen *et al*. [9, 10]).

where D_v denotes the deviation, U^* is the dimensionless velocity which is U_r/U_{tip} for the radial velocity or U_θ/U_{tip} for the tangential velocity. The subscripts "sim", "exp" and "max, exp" denote simulation result, experimental result, and maximum value of experimental results, respectively. Comparison of deviations between the statistics of 10 revolutions and 20 revolutions is shown in Tables V and VI, where $2r/D$ is the dimensionless radial position, $2z/w$ is the dimensionless axial position, "average" denotes the average of deviation, and the subscripts "20" and "10" denote the statistics for 20 revolutions and 10 revolutions, respectively. The results are given in Tables VII and VIII. The results suggest that there is a significant amount of statistical error in the results for 20 revolutions. This is especially true for positions near the tank wall. However, one may notice that the statistical results for 20 revolutions agree better with the experimental results. Therefore, the phase-resolved and phase-averaged results could be improved by increasing the statistics interval. As the resolution increases, the flow field tends to become more complex. Consequently, the statistics interval should be larger for a simulation with higher resolution. Eggels [3] suggested 40 revolutions for the statistics collection.

TABLE VI
Deviation for Tangential Velocity

		$2z/w$							
$2r/D$	Simulations	−1.6	−1.1	−0.55	0.0	0.55	1.1	1.6	Average
1.07	10 revolutions	0%	6%	15%	15%	1%	3%	26%	9.4%
	20 revolutions	1%	9%	18%	19%	1%	0%	24%	10.3%
	Derksen *et al.*'s simulation	6%	30%	24%	28%	36%	36%	0%	22.9%
1.29	10 revolutions	5%	20%	36%	38%	27%	23%	16%	23.5%
	20 revolutions	5%	23%	41%	24%	9%	27%	0%	18.2%
	Derksen *et al.*'s simulation	5%	11%	18%	15%	27%	45%	12%	19%
1.50	10 revolutions	24%	38%	47%	15%	10%	12%	5%	21.6%
	20 revolutions	24%	32%	32%	6%	10%	9%	18%	18.7%
	Derksen *et al.*'s simulation	9%	21%	21%	0%	13%	32%	27%	17.5%
1.66	10 revolutions	37%	57%	41%	16%	7%	5%	11%	24.9%
	20 revolutions	26%	50%	34%	5%	7%	5%	15%	20.3%

TABLE VII
Uncertainty for Radial Velocity

	$2z/w$						
$2r/D$	−1.6	−1.1	−0.55	0.0	0.55	1.1	1.6
1.07	100%	52%	7%	2%	3%	33%	60%
1.29	25%	0%	9%	8%	14%	0%	100%
1.50	4%	48%	15%	7%	13%	6%	0%
1.66	25%	43%	23%	8%	9%	0%	11%

TABLE VIII
Uncertainty for Tangential Velocity

	$2z/w$						
$2r/D$	−1.6	−1.1	−0.55	0.0	0.55	1.1	1.6
1.07	25%	17%	5%	3%	0%	33%	17%
1.29	0%	8%	11%	14%	20%	10%	233%
1.50	0%	14%	21%	9%	0%	5%	64%
1.66	27%	12%	9%	12%	0%	0%	70%

9. CONCLUSION

In this paper, large eddy simulations for the complex flow field in a stirred tank driven by a Rushton turbine have been investigated. The results show that the lattice Boltzmann scheme with the nonuniform grid technique and the arbitrary computational domain method is efficient for turbulent flow simulations that demand large computational resources. The nonuniform grid technique can be implemented efficiently in three-dimensional simulations on parallel computer platforms. One-dimensional quadratic interpolation applied in the nonuniform grid technique does not reduce the efficiency of the technique. Therefore, the application of this technique makes it feasible for the simulation of a high-resolution stirred tank started from a quiescent state. The phase-averaged results are comparable in accuracy to the results obtained by Eggels [3] and Derksen and Van den Akker [4]. The phase-resolved flow fields are also well predicted by the simulation. The deviation of these results from the experimental data is probably attributable to the fact that the subgrid-scale model does not incorporate the effects of the tank wall and to the short time interval used to collect statistics. The arbitrary computational domain method introduced in this paper has been shown to be useful in complex geometries. With this method, one can efficiently implement bounce-back boundary conditions with complex geometries.

ACKNOWLEDGMENTS

This work was supported by DuPont and by the Engineering Research Program of the Office of Basic Energy Science at the Department of Energy under Grant DE-FG02-88ER13919. We acknowledge the support and facilities of the National Center for Supercomputing Application at the University of Illinois at Urbana, Illinois.

REFERENCES

1. X. He, L.-S. Luo, and M. Dembo, Some progress in lattice Boltzmann method. Part 1. Nonuniform mesh grids, *J. Comput. Phys.* **129**, 357 (1996).
2. H. Wu and G. K. Patterson, Laser-Doppler measurements of turbulent-flow parameters in a stirred mixer, *Chem. Eng. Sci.* **44**, 2207 (1989).
3. J. G. M. Eggels, Direct and large-eddy simulation of turbulent flow using the lattice-Boltzmann scheme, *Int. J. Heat Fluid Flow* **17**, 307 (1996).
4. J. J. Derksen and H. E. A. Van den Akker, Large eddy simulations on the flow driven by a Rushton turbine, *AIChE J.* **45**, 209 (1999).
5. F. Nannelli and S. Succi, The lattice Boltzmann equation on irregular lattices, *J. Stat. Phys.* **68**, 401 (1992).
6. G. Amati, S. Succi, and R. Benzi, Turbulent channel flow simulation using a coarse-grained extension of the lattice Boltzmann method, *Fluid Dyn. Res.* **19**, 289 (1997).
7. O. Filippova and D. Hänel, Grid refinement for lattice-BGK models, *J. Comput. Phys.* **147**, 219 (1998).
8. X. He and G. D. Doolen, Lattice Boltzmann method on curvilinear coordinates system: Flow around a circular cylinder, *J. Comput. Phys.* **134**, 306 (1997).
9. J. J. Derksen, M. S. Doelman, and H. E. A. Van den Akker, Phase-resolved three-dimensional LDA measurements in the impeller region of a turbulently stirred tank, in *International Symposium on Applications of Laser Techniques to Fluid Mechanics, Lisbon*, 1998, p. 233.
10. J. J. Derksen, M. S. Doelman, and H. E. A. Van den Akker, Three-dimensional LDA measurements in the impeller region of a turbulently stirred tank, *Exp. Fluids* **27**, 522 (1999).
11. U. Frisch, B. Hasslacher, and Y. Pomeau, Lattice-gas automana for the Navier-Stokes equations, *Phys. Rev. Lett.* **56**, 1505 (1986).
12. S. Chen and G. D. Doolen, Lattice Boltzmann method for fluid flows, *Annu. Rev. Fluid Mech.* **30**, 329 (1998).

13. D. d'Humieres, P. Lallemand, and U. Frisch, Lattice gas models for 3-D hydrodynamics, *Europhys. Lett.* **2**, 291 (1986).
14. J. G. M. Eggels and J. A. Somers, Numerical simulation of free convective flow using the lattice-Boltzmann scheme, *Int. J. Heat Fluid Flow* **16**, 357 (1995).
15. W. Gropp, E. Lusk, and A. Skjellum, *Using MPI Portable Parallel Programming with the Message-Passing Interface* (MIT Press, Cambridge, MA, 1994).
16. A. J. C. Ladd, Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1. Theoretical foundation, *J. Fluid Mech.* **271**, 285 (1994).
17. J. Smagorinsky, General circulation experiments with the primitive equations: I. The basic equations, *Mon. Weather Rev.* **91**, 99 (1963).
18. J. A. Somers, Direct simulation of fluid flow with cellular automata and the lattice-Boltzmann equation, *Appl. Sci. Res.* **51**, 127 (1993).
19. U. Piomelli, P. Moin, and J. H. Ferziger, Model consistency in large eddy simulation of turbulence channel flows, *Phys. Fluids* **31**, 1884 (1988).
20. J. O. Hinze, Fundamentals of the hydrodynamic mechanism of splitting in dispersion processes, *AIChE J.* **1**, 289 (1955).
21. M. J. Prince and H. W. Blanch, Bubble coalescence and break-up in air-sparged bubble columns, *AIChE J.* **36**, 1485 (1990).