Comput. Methods Appl. Mech. Engrg. 257 (2013) 36–46

Contents lists available at SciVerse ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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

Review

Three-dimensional volume-conserving immersed boundary model for two-phase fluid flows

Yibao Li, Ana Yun, Dongsun Lee, Jaemin Shin, Darae Jeong, Junseok Kim *

Department of Mathematics, Korea University, Seoul 136-713, Republic of Korea

ARTICLE INFO

Article history:
Received 27 July 2012
Received in revised form 15 January 2013
Accepted 18 January 2013
Available online 29 January 2013

Keywords:
Immersed boundary method
Volume-preserving
Two-phase fluid flow
Multigrid method
Finite difference

ABSTRACT

We present a volume-preserving scheme for two-phase immiscible incompressible flows using an immersed boundary method (IBM) in a three-dimensional space. The two-phase IBM employs a mixture of Eulerian and Lagrangian variables, where the fluid interface is represented by discrete Lagrangian markers exerting surface tension forces to the Eulerian fluid domain and the markers are advected by the fluid velocity. The interactions between the Lagrangian markers and the fluid variables are linked by the discretized Dirac delta function. The present study extends the previous two-dimensional research (Li et al., Volume preserving immersed boundary methods for two-phase fluid flows, Int. J. Numer. Meth. Fluids 69 (2012) 842–858) to the three-dimensional space. The key idea of the proposed method is relocating surface points along the normal directions to conserve the total volume. We perform a number of numerical experiments to show the efficiency and accuracy of the proposed method.

Contents

1. Introduction

Two-phase fluid flows have been studied by using various methods such as the volume of fluid [1–3], level set [4–7], front tracking [8–10], phase field [11–13], and lattice Boltzmann methods [14,15]. As the immersed boundary method (IBM) proposed by Peskin [16] can accurately simulate the interaction between thin elastic material and fluid, it has been widely studied and applied successfully to many biological problems [17–29]. The main idea of the IBM is to use an Eulerian grid for the fluid mechanics together with Lagrangian markers of the immersed boundary in the fluid. The immersed boundary exerts a singular force on the fluid and moves at the fluid velocity. The interaction between the fluid and the immersed boundary is achieved through the spread-

[*] Corresponding author. Tel.: +82 2 3290 3077; fax: +82 2 929 8562.
E-mail address: cfdkim@korea.ac.kr (J. Kim).
URL: http://math.korea.ac.kr/~cfdkim/ (J. Kim).

0045-7825/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cma.2013.01.009

ing of the singular force and the interpolation of the velocity by using a discrete Dirac delta function [30,31].

Though the IBM has been studied for many applications including the flow in the heart [16], the IBM does not conserve the volume generally [32] and it is a drawback for simulating immiscible multi-phase fluid flows. The volume loss can be caused by the violated divergence-free condition at Lagrangian grid of the immersed boundary point as the interpolated velocity field through the delta function cannot be guaranteed to be continuously divergence-free [33,34]. To resolve this problem, Peskin and Printz [34] proposed the modified finite-difference divergence stencils based on the interpolation scheme. The proposed method is constructed in such a way that the interpolated velocity field at the immersed boundary is more nearly divergence-free. However, this method does not guarantee the conservation of the polyhedron volume since we use a finite time step size to advect the immersed boundary marker points.

In this paper, we present a simple volume-preserving scheme for two-phase immiscible incompressible fluid flows using the IBM in a three-dimensional space, which is an extension of the work of Li et al. [35] in which they focused on an area-preserving scheme in a two-dimensional space. The key idea is to relocate the interface marker points normally so that the polyhedron volume is preserved. Recently, some papers are reported on an improved mass conservation of finite element tests [36,37]. The severity of volume loss depends on many factors such as the grid structure, the time integrator, and the constitutive law for the boundary mechanics. It is worth noting that the lack of volume conservation is not a problem for many IBM applications such as those for which the structure has stress-free reference configuration.

The rest of the paper is organized as follows. Section 2 briefly summarizes the IBM for the immiscible incompressible two-phase fluid flows. In Section 3, the numerical solutions including the surface mesh generation, calculation of mean curvature, and volume correction algorithm are described. Representative numerical experiments are provided to show the efficiency and accuracy of our volume-preserving scheme in Section 4. Finally, conclusions are drawn in Section 5.

2. Governing equations

Let the fluid velocity $\mathbf{u}(\mathbf{x},t) = (u(\mathbf{x},t), v(\mathbf{x},t), w(\mathbf{x},t))$ be defined on the fixed Cartesian coordinate $\mathbf{x} = (x,y,z)$ at time t (see Fig. 1). Viscous and incompressible fluid flows in a Cartesian domain $\Omega \in \mathbb{R}^3$ containing an immersed boundary Γ can be modeled by the Navier–Stokes equation:

$$\rho(I)\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\Delta\mathbf{u} + \mathbf{f} + \rho(I)\mathbf{g}, \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{2}$$

where $I(\mathbf{x},t)$ is the indicator function, $\rho(I(\mathbf{x},t))$ is the fluid density, $p(\mathbf{x},t)$ is the pressure, μ is the viscosity, and $\mathbf{g}$ is the gravity. Here, the singular surface tension force density $\mathbf{f}(\mathbf{x},t)$ is given by

$$\mathbf{f}(\mathbf{x},t) = \int_\Gamma \mathbf{F}(\mathbf{X}(t))\delta(\mathbf{x}-\mathbf{X}(t))d\mathcal{A}, \tag{3}$$

where $\mathbf{X}(t)$ is the Lagrangian variable for the immersed boundary (see Fig. 1) and $\delta(\mathbf{x}-\mathbf{X}(t))$ is the three-dimensional Dirac delta function which is defined by the product of one-dimensional Dirac delta functions, i.e., $\delta(\mathbf{x}) = \delta(x)\delta(y)\delta(z)$. The Lagrangian force density $\mathbf{F}(\mathbf{X}(t))$ is defined as

$$\mathbf{F}(\mathbf{X}(t)) = \sigma\boldsymbol{\kappa}(\mathbf{X}(t))\mathbf{n}(\mathbf{X}(t)), \tag{4}$$

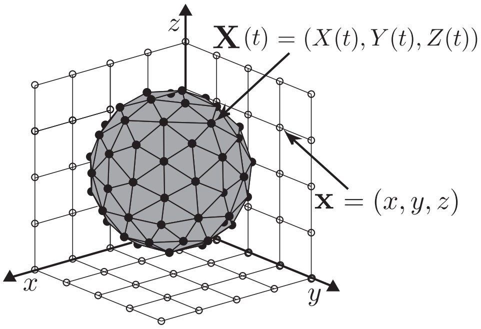

Fig. 1. Illustration of Eulerian points $\mathbf{x}$ and Lagrangian points $\mathbf{X}(t)$.

where σ is the surface tension coefficient, $\boldsymbol{\kappa} = \boldsymbol{\kappa}_1 + \boldsymbol{\kappa}_2$ is the mean curvature for the two principal curvatures $\boldsymbol{\kappa}_1,\ \boldsymbol{\kappa}_2$, and $\mathbf{n}$ is the unit outward normal vector.

The indicator function is obtained by solving the following Poisson equation with zero Dirichlet boundary condition:

$$\Delta I(\mathbf{x},t) = -\nabla\cdot\int_\Gamma \mathbf{n}(\mathbf{X}(t))\delta(\mathbf{x}-\mathbf{X}(t))d\mathcal{A}.$$

The variable density is calculated by $\rho(I(\mathbf{x},t)) = \rho_2 + (\rho_1-\rho_2)I(\mathbf{x},t)$, where ρ_1 and ρ_2 are densities of fluid 1 and fluid 2, respectively. The immersed boundary velocity $\mathbf{U}(\mathbf{X}(t))$ is determined by

$$\mathbf{U}(\mathbf{X}(t)) = \int_\Omega \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(t))d\mathbf{x}. \tag{5}$$

Then the motion of the immersed boundary is determined by integrating the equation

$$\frac{d\mathbf{X}(t)}{dt} = \mathbf{U}(\mathbf{X}(t)). \tag{6}$$

3. Numerical solution

In this section, we present the numerical method for solving the governing equations.

3.1. Numerical algorithm

Let a computational domain be partitioned with a uniform mesh spacing h in Cartesian geometry. The center of each cell is located at $\mathbf{x}_{ijk} = (x_i, y_j, z_k) = ((i-0.5)h, (j-0.5)h, (k-0.5)h)$ for $i = 1,\ldots,N_x$, $j = 1,\ldots,N_y$, and $k = 1,\ldots,N_z$. Here, N_x, N_y, and N_z are the numbers of cells in the x-, y- and z-directions, respectively. We denote the final time by T and the time step by Δt.

To generate an oriented triangular mesh connecting with immersed boundary points, we use the distmesh algorithm [38,39]. Let us take a two-dimensional example to illustrate the basic idea of the algorithm. Let the interface of two immiscible fluids be implicitly represented as the zero level set of a signed distance function. For example, for the unit circle interface, a signed distance function is given as $\phi(x,y) = \sqrt{x^2+y^2} - 1$ (Fig. 2a). Here are several steps on generating the triangular mesh.

Step A. Create an initial distribution in the computational domain (Fig. 2b).
Step B. Remove points outside the region, $\phi(x,y) > 0$ (Fig. 2c).
Step C. Make triangulation by the Delaunay algorithm [40,41] (Fig. 2c).
Step D. Assign the repulsive force depending on the length of side of each triangle and calculate the net force at each node point (Fig. 2d). Move points according to the net force.

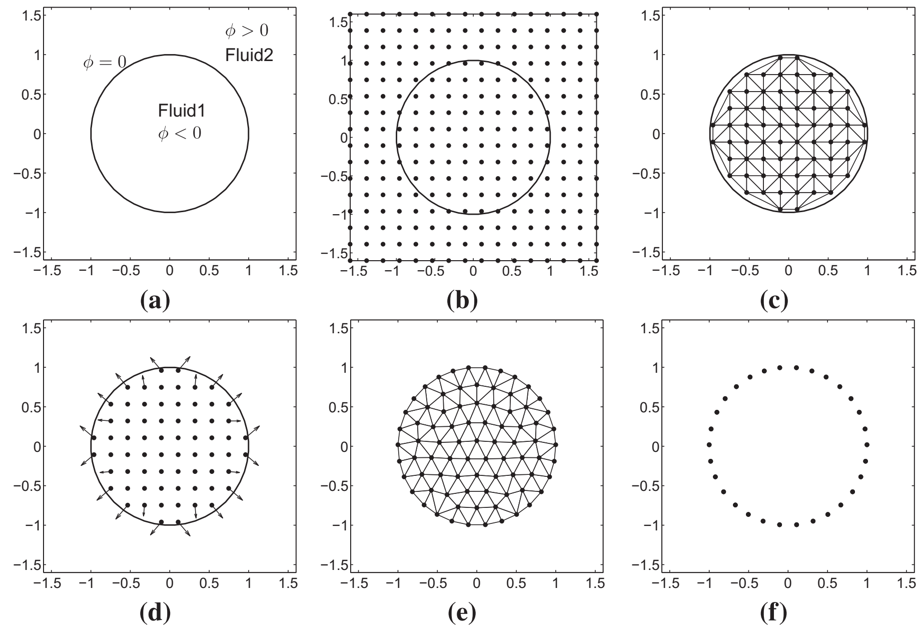

Fig. 2. (a) Implicit representation of the interface, (b) initial distribution, (c) triangulation after removing outside points, (d) net forces, (e) final mesh, and (f) boundary points.

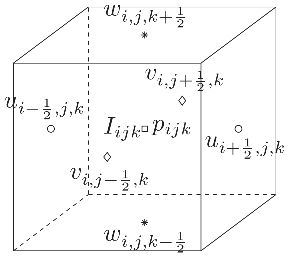

Fig. 3. Velocities are defined at cell boundaries while the pressure and indicator function are defined at the cell centers.

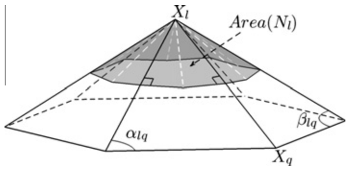

Fig. 4. Schematic of the Voronoi area.

Step E. Find points outside $\phi(x,y) > 0$ and bring them back to the boundary $\phi(x,y) = 0$.

We repeat *Steps C–E* until the mesh quality is satisfactory, that is, all node points move less than a given tolerance (see [38,39]) (Fig. 2e), then take boundary points as the immersed boundary markers (Fig. 2f).

We use a set of M Lagrangian points $\mathbf{X}_l = (X_l, Y_l, Z_l)$ for $l = 1, \ldots, M$ to represent the immersed boundary. Suppose that there are M_T triangles $\mathbf{Tri}_s = (\mathbf{X}_l, \mathbf{X}_m, \mathbf{X}_q)$ for $s = 1, \ldots, M_T$. Note that the three vertices $\mathbf{X}_l$, $\mathbf{X}_m$, and $\mathbf{X}_q$ are ordered counterclockwise.

Let us discretize the governing equations. A staggered marker-and-cell (MAC) mesh of Harlow and Welch [42] is used. Pressure and indicator functions are stored at cell centers. Velocity components u, v, and w are defined at the x-, y-, and z-directional face centers, respectively (see Fig. 3). That is, $u^n_{i+\frac{1}{2}jk} = u(x_{i+\frac{1}{2}}, y_j, z_k, n\Delta t)$, $v^n_{i,j+\frac{1}{2},k} = (x_i, y_{j+\frac{1}{2}}, z_k, n\Delta t)$, and $w^n_{ij,k+\frac{1}{2}} = (x_i, y_j, z_{k+\frac{1}{2}}, n\Delta t)$.

Next, we describe the overall procedure for the numerical solution from n to $n+1$ time step. With a given divergence-free velocity field $\mathbf{u}^n$ and a boundary configuration $\mathbf{X}^n$, we want to find $\mathbf{u}^{n+1}$ and $\mathbf{X}^{n+1}$ which are solved using the following fractional step method:

$$\rho^n \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = -\rho^n \mathbf{u}^n \cdot \nabla_d \mathbf{u}^n - \nabla_d p^{n+1} + \mu \Delta_d \mathbf{u}^n + \mathbf{f}^n + \rho^n \mathbf{g}, \tag{7}$$

$$\nabla_d \cdot \mathbf{u}^{n+1} = 0, \tag{8}$$

where $\rho^n = \rho_2 + (\rho_1 - \rho_2) I^n$ and $\mathbf{g} = (0, 0, -g)$.

Step 1. Evaluate the surface tension force $\mathbf{F}^n$ on the immersed boundary from the given boundary configuration $\mathbf{X}^n$. For $l = 1, \ldots, M$,

$$\mathbf{F}^n_l = \sigma \boldsymbol{\kappa}^n_l \mathbf{n}^n_l. \tag{9}$$

We briefly review the calculation of the normal mean curvature at the immersed boundary point on the surface [43–48]. Let us denote a Voronoi area at $\mathbf{X}_l$ by $Area(N_l)$, where N_l is the set of neighboring vertices of $\mathbf{X}_l$. That is,

$$Area(N_l) = \frac{1}{8} \sum_{q \in N_l} (\cot \alpha_{lq} + \cot \beta_{lq}) |\mathbf{X}_l - \mathbf{X}_q|^2,$$

where α_{lq} and β_{lq} are angles opposite to the edge $\overline{\mathbf{X}_l \mathbf{X}_q}$ (see Fig. 4). Then the normal mean curvature is given by

$$\begin{aligned} \kappa(\mathbf{X}_l)\mathbf{n} &= \lim_{Area(N_l) \to 0} \frac{\nabla Area(N_l)}{Area(N_l)} \\ &= \frac{1}{2Area(N_l)} \sum_{q \in N_l} (\cot \alpha_{lq} + \cot \beta_{lq})(\mathbf{X}_l - \mathbf{X}_q), \end{aligned}$$

where ∇ is the gradient with respect to the (x, y, z) coordinates of $\mathbf{X}_l$ [43].

To test this formula for computing the normal mean curvature, we consider a unit sphere placed at the center of domain $\Omega = (-1, 1) \times (-1, 1) \times (-1, 1)$. Then the analytic value of the mean curvature is $\kappa(\mathbf{X}) = 2$ for any point $\mathbf{X}$ on the surface.

First, we consider numerical mean curvature values with non-obtuse and obtuse triangle meshes as shown in Fig. 5: (a) non-obtuse triangles, the numerical value of κ is 2, which means the error

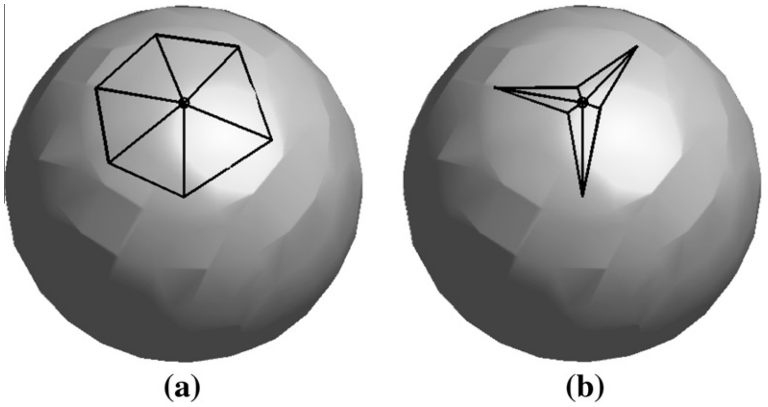

Fig. 5. Mean curvature calculated by various triangles in a unit sphere with analytic value $\kappa = 2$. (a) Non-obtuse triangles, the numerical value of κ is 2, which means the error is on the order of machine precision. (b) Obtuse triangles, the numerical value of κ is 2.00002.

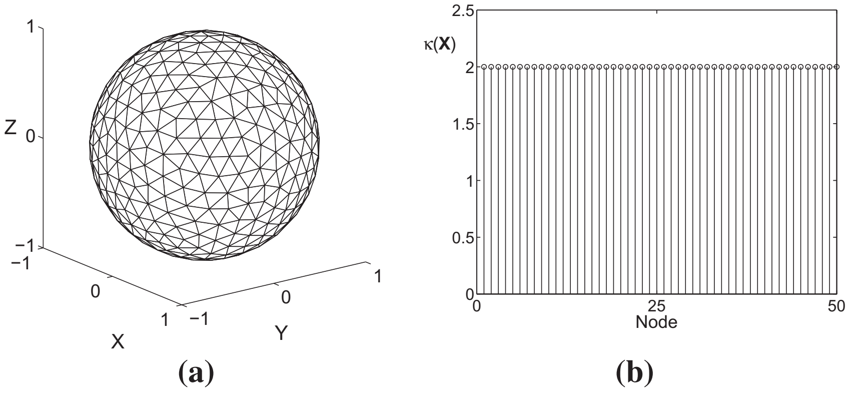

Fig. 6. (a) Triangular mesh generation of the unit sphere. (b) Stem plot of the normal mean curvature $\kappa(\mathbf{X})$ for 50 randomly chosen surface points.

is on the order of machine precision and (b) obtuse triangles, the numerical value of κ is 2.00002. We have almost identical numerical results for both cases comparing with the analytic mean curvature value.

Second, Fig. 6a and b shows the triangular mesh of the unit sphere by using the distmesh algorithm and the stem plot of $\kappa(\mathbf{X})$ for 50 randomly chosen surface points, respectively. In this case, we calculate the discrete l_2-norm of error, which is $\sqrt{\frac{1}{M}\sum_{l=1}^{M}|2-\kappa(\mathbf{X}_l)|^2} = 3.02\text{e}{-5}$. As can be seen from these results, the formula is accurate.

Step 2. Spread the boundary surface tension force to the nearby lattice points of the fluid.

$$\mathbf{f}^n_{ijk} = \sum_{l=1}^{M}\mathbf{F}^n_l\delta_h(\mathbf{x}_{ijk}-\mathbf{X}^n_l)\Delta\mathcal{A}_l, \quad \text{for } i=1,\ldots,N_x, \quad j=1,\ldots,N_y, \quad k = 1,\ldots,N_z.$$

Here δ_h is a smoothed approximation to the three-dimensional Dirac delta function as

$$\delta_h(\mathbf{x}) = \frac{1}{h^3}\delta\left(\frac{x}{h}\right)\delta\left(\frac{y}{h}\right)\delta\left(\frac{z}{h}\right),$$

where a 4-point delta function [30] is given by

$$\delta(r) = \begin{cases} (3-2|r|+\sqrt{1+4|r|-4r^2})/8 & \text{if } |r| \leqslant 1, \\ (5-2|r|-\sqrt{-7+12|r|-4r^2})/8 & \text{if } 1<|r|\leqslant 2, \\ 0 & \text{if } |r|>2. \end{cases} \tag{10}$$

Here the surface area element $\Delta\mathcal{A}_l$ is defined as one third of the total surface area of neighboring triangles having $\mathbf{X}_l$ as the common vertex, i.e.,

$$\Delta\mathcal{A}_l = \frac{1}{6}\sum_{m,q\in N_l}\sqrt{|\mathbf{X}_l-\mathbf{X}_m|^2|\mathbf{X}_l-\mathbf{X}_q|^2-((\mathbf{X}_l-\mathbf{X}_m)\cdot(\mathbf{X}_l-\mathbf{X}_q))^2}.$$

Step 3. Solve the Navier–Stokes Eq. (7) on the Cartesian grid to get $\boldsymbol{u}^{n+1}$ and p^{n+1} from $\boldsymbol{u}^n$, $\boldsymbol{f}^{\,n} = (f^n_1, f^n_2, f^n_3)$, and $\mathbf{g} = (g_1, g_2, g_3)$:

$$\rho^n_{i+\frac{1}{2},jk}\left(\frac{u^{n+1}-u^n}{\Delta t}+\mathbf{u}^n\cdot\nabla_d u^n\right)_{i+\frac{1}{2},jk} = -\frac{p^{n+1}_{i+1,jk}-p^{n+1}_{ijk}}{h}+\mu\Delta_d u^n_{i+\frac{1}{2},jk}+f^n_{1\,i+\frac{1}{2},jk}+\rho^n_{i+\frac{1}{2},jk}g_1, \tag{11}$$

$$\rho^n_{i,j+\frac{1}{2},k}\left(\frac{v^{n+1}-v^n}{\Delta t}+\mathbf{u}^n\cdot\nabla_d v^n\right)_{i,j+\frac{1}{2},k} = -\frac{p^{n+1}_{i,j+1,k}-p^{n+1}_{ijk}}{h}+\mu\Delta_d v^n_{i,j+\frac{1}{2},k}+f^n_{2\,i,j+\frac{1}{2},k}+\rho^n_{i,j+\frac{1}{2},k}g_2, \tag{12}$$

$$\rho^n_{ij,k+\frac{1}{2}}\left(\frac{w^{n+1}-w^n}{\Delta t}+\mathbf{u}^n\cdot\nabla_d w^n\right)_{ij,k+\frac{1}{2}} = -\frac{p^{n+1}_{ij,k+1}-p^{n+1}_{ijk}}{h}+\mu\Delta_d w^n_{ij,k+\frac{1}{2}}+f^n_{3\,ij,k+\frac{1}{2}}+\rho^n_{ij,k+\frac{1}{2}}g_3, \tag{13}$$

$$\nabla_d\cdot\boldsymbol{u}^{n+1}_{ijk} = 0. \tag{14}$$

By using the projection method proposed by Chorin [49], compute an intermediate velocity field, $\tilde{\boldsymbol{u}}$, which does not satisfy the incompressible condition. For example, from Eq. (11) without the pressure gradient term, we have

$$\rho^n_{i+\frac{1}{2}jk}\left(\frac{\tilde{u}^{n+1}-u^n}{\Delta t}+\mathbf{u}^n\cdot\nabla_d u^n\right)_{i+\frac{1}{2}jk}=\mu\Delta_d u^n_{i+\frac{1}{2}jk}+f^n_{1\ i+\frac{1}{2}jk}+\rho^n_{i+\frac{1}{2}jk}g_1.$$

Then we have

$$\begin{aligned}\tilde{u}_{i+\frac{1}{2}jk}&=u^n_{i+\frac{1}{2}jk}-\Delta t(\mathbf{u}^n\cdot\nabla_d u^n)_{i+\frac{1}{2}jk}+\frac{\Delta t}{\rho^n_{i+\frac{1}{2}jk}}f^n_{1\ i+\frac{1}{2}jk}\\&+\frac{\Delta t\mu}{h^2\rho^n_{i+\frac{1}{2}jk}}\Big(u^n_{i+\frac{3}{2}jk}+u^n_{i-\frac{1}{2}jk}+u^n_{i+\frac{1}{2}j+1,k}+u^n_{i+\frac{1}{2}j-1,k}+u^n_{i+\frac{1}{2}j,k+1}\\&\quad++u^n_{i+\frac{1}{2}j,k-1}-6u^n_{i+\frac{1}{2}jk}\Big)+g_1\,\Delta t,\end{aligned}\tag{15}$$

where the advection term is defined by

$$\begin{aligned}(\mathbf{u}^n\cdot\nabla_d u^n)_{i+\frac{1}{2}jk}&=u^n_{i+\frac{1}{2}jk}\bar{u}^n_{x_{i+\frac{1}{2}jk}}\\&+\frac{v^n_{i,j-\frac{1}{2},k}+v^n_{i+1,j-\frac{1}{2},k}+v^n_{i,j+\frac{1}{2},k}+v^n_{i+1,j+\frac{1}{2},k}}{4}\bar{u}^n_{y_{i+\frac{1}{2}jk}}\\&+\frac{w^n_{ij,k-\frac{1}{2}}+w^n_{i+1,j,k-\frac{1}{2}}+w^n_{ij,k+\frac{1}{2}}+w^n_{i+1,j,k+\frac{1}{2}}}{4}\bar{u}^n_{z_{i+\frac{1}{2}jk}}.\end{aligned}\tag{16}$$

The values $\bar{u}^n_{x_{i+\frac{1}{2}jk}}$ is computed using the upwind method defined as follows:

$$\bar{u}^n_{x_{i+\frac{1}{2}jk}}=\begin{cases}\frac{u^n_{i+\frac{1}{2}jk}-u^n_{i-\frac{1}{2}jk}}{h} & \text{if } u^n_{i+\frac{1}{2}jk}>0,\\ \frac{u^n_{i+\frac{3}{2}jk}-u^n_{i+\frac{1}{2}jk}}{h} & \text{otherwise.}\end{cases}\tag{17}$$

The other values $\bar{u}^n_{y_{i+\frac{1}{2}jk}}$ and $\bar{u}^n_{z_{i+\frac{1}{2}jk}}$ are calculated similarly as above. Then, we solve the following equations for the advanced pressure field at $(n+1)$ time step.

$$\rho^n\frac{\boldsymbol{u}^{n+1}-\tilde{\boldsymbol{u}}}{\Delta t}=-\nabla_d p^{n+1},\tag{18}$$

$$\nabla_d\cdot\boldsymbol{u}^{n+1}=0.\tag{19}$$

Applying the divergence operator to Eq. (18), we have the Poisson equation for the pressure at the advanced time $(n+1)$.

$$\nabla_d\cdot\left(\frac{1}{\rho^n}\nabla_d p^{n+1}\right)=\frac{1}{\Delta t}\nabla_d\cdot\tilde{\boldsymbol{u}},\tag{20}$$

where

$$\begin{aligned}\nabla_d\cdot\left(\frac{1}{\rho^n}\nabla_d p^{n+1}_{ijk}\right)&=\frac{\frac{p^{n+1}_{i+1,jk}}{\rho^n_{i+\frac{1}{2}jk}}+\frac{p^{n+1}_{i-1,jk}}{\rho^n_{i-\frac{1}{2}jk}}+\frac{p^{n+1}_{i,j+1,k}}{\rho^n_{i,j+\frac{1}{2},k}}+\frac{p^{n+1}_{i,j-1,k}}{\rho^n_{i,j-\frac{1}{2},k}}+\frac{p^{n+1}_{ij,k+1}}{\rho^n_{ij,k+\frac{1}{2}}}+\frac{p^{n+1}_{ij,k-1}}{\rho^n_{ij,k-\frac{1}{2}}}}{h^2}\\&-\frac{\frac{1}{\rho^n_{i+\frac{1}{2}jk}}+\frac{1}{\rho^n_{i-\frac{1}{2}jk}}+\frac{1}{\rho^n_{i,j+\frac{1}{2},k}}+\frac{1}{\rho^n_{i,j-\frac{1}{2},k}}+\frac{1}{\rho^n_{ij,k+\frac{1}{2}}}+\frac{1}{\rho^n_{ij,k-\frac{1}{2}}}}{h^2}p^{n+1}_{ijk},\end{aligned}\tag{21}$$

$$\nabla_d\cdot\tilde{\mathbf{u}}_{ijk}=\frac{\tilde{u}_{i+\frac{1}{2}jk}-\tilde{u}_{i-\frac{1}{2}jk}}{h}+\frac{\tilde{v}_{i,j+\frac{1}{2},k}-\tilde{v}_{i,j-\frac{1}{2},k}}{h}+\frac{\tilde{w}_{ij,k+\frac{1}{2}}-\tilde{w}_{ij,k-\frac{1}{2}}}{h}.\tag{22}$$

The linear system of Eq. (20) is solved using a multigrid method [50], specifically, V-cycles using a Gauss–Seidel relaxation with a tolerance 1e−7. After solving the pressure field, we update $u^{n+1}_{i+\frac{1}{2},j,k}$ as

$$u^{n+1}_{i+\frac{1}{2}jk}=\tilde{u}_{i+\frac{1}{2}jk}-\frac{\Delta t}{\rho^n_{i+\frac{1}{2}jk}}\frac{p^{n+1}_{i+1,jk}-p^{n+1}_{ijk}}{h}.\tag{23}$$

The variables $v^{n+1}_{i,j+\frac{1}{2},k}$ and $w^{n+1}_{ij,k+\frac{1}{2}}$ are updated in a similar manner.

Step 4. Using the updated fluid velocity $\mathbf{u}^{n+1}$, we evaluate the immersed boundary velocity $\mathbf{U}^{n+1}$ and then the new boundary position $\mathbf{X}^{n+1}$ is updated according to

$$\mathbf{U}^{n+1}_l=\sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\sum_{k=1}^{N_z}\mathbf{u}^{n+1}_{ijk}\delta_h(\mathbf{x}_{ijk}-\mathbf{X}^n_l)h^3,\tag{24}$$

$$\mathbf{X}^{n+1}_l=\mathbf{X}^n_l+\Delta t\mathbf{U}^{n+1}_l\text{for } l=1,\ldots,M.\tag{25}$$

This completes the process (*Step 1–Step 4*) by which the variables $\mathbf{u}^{n+1}$ and $\mathbf{X}^{n+1}$ are calculated.

3.2. Volume correction algorithm

In this section, we describe the volume correction algorithm. First, let us define the volume of the polyhedron. We take a triangle on the surface $\mathbf{Tri}_s=(\mathbf{X}_l,\mathbf{X}_m,\mathbf{X}_q)=((X_l,Y_l,Z_l),(X_m,Y_m,Z_m),(X_q,Y_q,Z_q))$ with a reference point $(\mathbf{O})$ (see Fig. 7) and calculate the volume of tetrahedra one by one. $\mathbf{Tri}_s$ has positive orientation since $\mathbf{X}_l,\mathbf{X}_m$, and $\mathbf{X}_q$ are in counterclockwise order.

Then we obtain the following formula for the volume of the polyhedron

$$V(\mathbf{X})=\frac{1}{6}\sum_{s=1}^{M_T}\left[X_q(Y_lZ_m-Y_mZ_l)-Y_q(X_lZ_m-X_mZ_l)+Z_q(X_lY_m-X_mY_l)\right],$$

where M_T is the number of surface triangles [51–55]. This volume formula is independent of the location of the reference point and is valid even for a nonconvex polyhedron [51,52]. We define the relative error of volume $V(\mathbf{X})$ with the initial volume $V(\mathbf{X}^0)$ as

$$V_{error}(\mathbf{X})=\frac{|V(\mathbf{X}^0)-V(\mathbf{X})|}{V(\mathbf{X}^0)}.\tag{26}$$

To numerically demonstrate the invariant of the volume formula on the reference point locations and the convexity of the surface, we consider a torus as a test surface. The torus in Fig. 8 is the nonconvex geometry and the origin is not inside the geometry. For generating the surface mesh of the torus, we use the usual parametrization of the torus,

$$\mathbf{T}(\theta,\phi)=((3+\cos\theta)\cos\phi,(3+\cos\theta)\sin\phi,\sin\theta),\quad 0\leqslant\theta,\phi\leqslant 2\pi.$$

To generate the mesh, we divide the domain into $\theta_i=2\pi i/N_{torus}$, $\phi_j=2\pi j/N_{torus}$, for $0\leqslant i,\ j<N_{torus}$. Then $\mathbf{T}(\theta_i,\ \phi_j)$ becomes the surface point, see Fig. 8. The computed volumes are 57.76, 58.88, 59.14, and 59.20 with $N_{torus}=25,50,100$, and 200, respectively. This result shows that the computed volume converges to the analytic volume value $6\pi^2\approx 59.22$ as we increase the number of surface points.

The outline of the volume correction procedure is:

Step (1) Update the immersed boundary points X^*_l, Y^*_l, and Z^*_l according to Eq. (25)

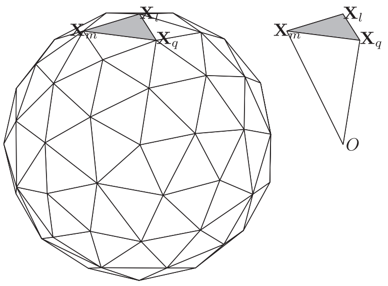

Fig. 7. Schematic illustration for the polyhedron and a tetrahedron with a reference point **O**.

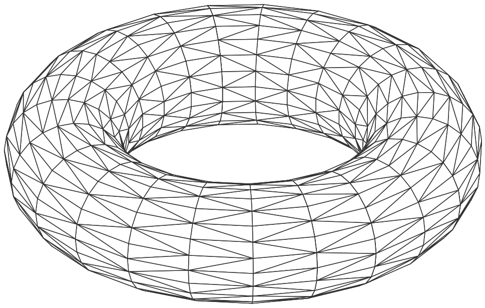

Fig. 8. Torus surface mesh with $N_{torus} = 25$.

$$(X_l^*, Y_l^*, Z_l^*) = (X_l^n, Y_l^n, Z_l^n) + \Delta t(U_l^{n+1}, V_l^{n+1}, W_l^{n+1}). \tag{27}$$

Step (2) Compute the polyhedral volume $V(\mathbf{X}^*)$

For a given tolerance *tol*, we check whether $V_{error}(\mathbf{X}^*) < tol$ or not. If it is true, then $\mathbf{X}^{n+1} = \mathbf{X}^*$. If not, update the interface X_l^{n+1}, Y_l^{n+1}, and Z_l^{n+1} with the following Eq. (28) by taking the volume correction algorithm, *Step 3)*.

$$(X_l^{n+1}, Y_l^{n+1}, Z_l^{n+1}) = (X_l^*, Y_l^*, Z_l^*) + \epsilon(\alpha_l, \beta_l, \gamma_l). \tag{28}$$

Here, $(\alpha_l, \beta_l, \gamma_l)$ is the outward unit normal vector at the lth interface node at time level t^* and ϵ is a constant, which will be determined in the next step.*Step (3) Determine the parameter ϵ*

The parameter ϵ is a root of the following cubic equation which is obtained from the volume correction, i.e., $V(\mathbf{X}^{n+1}) = V(\mathbf{X}^* + \epsilon\mathbf{n}) = V(\mathbf{X}^0)$:

$$V(\mathbf{X}^0) = V(\mathbf{X}^*) + a\epsilon^3 + b\epsilon^2 + c\epsilon, \tag{29}$$

where

$$a = \frac{1}{6}\sum_{s=1}^{M_T}\Big[\gamma_q(\alpha_l\beta_m - \beta_l\alpha_m) - \gamma_m(\alpha_l\beta_q - \beta_l\alpha_q) + \gamma_l(\alpha_m\beta_q - \beta_m\alpha_q)\Big], \tag{30}$$

$$\begin{aligned} b = \frac{1}{6}\sum_{s=1}^{M_T}\Big[&X_l^*(\beta_m\gamma_q - \gamma_m\beta_q) - Y_l^*(\alpha_m\gamma_q - \gamma_m\alpha_q) + Z_l^*(\alpha_m\beta_q - \beta_m\alpha_q) \\ &- X_m^*(\beta_l\gamma_q - \gamma_l\beta_q) + Y_m^*(\alpha_l\gamma_q - \gamma_l\alpha_q) - Z_m^*(\alpha_l\beta_q - \beta_l\alpha_q) \\ &+ X_q^*(\beta_l\gamma_m - \gamma_l\beta_m) - Y_q^*(\alpha_l\gamma_m - \gamma_l\alpha_m) + Z_q^*(\alpha_l\beta_m - \beta_l\alpha_m)\Big], \end{aligned} \tag{31}$$

$$\begin{aligned} c = \frac{1}{6}\sum_{s=1}^{M_T}\Big[&\alpha_l\Big(Y_m^*Z_q^* - Z_m^*Y_q^*\Big) - \beta_l\Big(X_m^*Z_q^* - Z_m^*X_q^*\Big) + \gamma_l\Big(X_m^*Y_q^* - Y_m^*X_q^*\Big) \\ &- \alpha_m(Y_l^*Z_q^* - Z_l^*Y_q^*) + \beta_m\Big(X_l^*Z_q^* - Z_l^*X_q^*\Big) - \gamma_m\Big(X_l^*Y_q^* - Y_l^*X_q^*\Big) \\ &+\alpha_q(Y_l^*Z_m^* - Z_l^*Y_m^*) - \beta_q(X_l^*Z_m^* - Z_l^*X_m^*) + \gamma_q(X_l^*Y_m^* - Y_l^*X_m^*)\Big]. \end{aligned} \tag{32}$$

Here, $(\alpha_l, \beta_l, \gamma_l), (\alpha_m, \beta_m, \gamma_m)$, and $(\alpha_q, \beta_q, \gamma_q)$ are corresponding normal vectors at each point $(X_l^*, Y_l^*, Z_l^*), (X_m^*, Y_m^*, Z_m^*)$, and $\Big(X_q^*, Y_q^*, Z_q^*\Big)$, respectively.

If $V(\mathbf{X}^*) > V(\mathbf{X}^0)$, then ϵ should be negative to lower the volume. In the same way, if $V(\mathbf{X}^*) < V(\mathbf{X}^0)$, then $\epsilon > 0$. To find a root of Eq. (29), we use the Newton's method with a zero initial guess, $\epsilon = 0$. The residual error converges rather quickly to a tolerance 1e−7 in 3–5 iterations.

Fig. 9 shows examples of volume corrections for (a) a sphere and (b) an ellipsoid. The first column is configurations before corrections with $V(\mathbf{X}^*) = 0.524$. The second and third columns are after corrections in the cases of $V(\mathbf{X}^0) = 0.262 < V(\mathbf{X}^*)$ and $V(\mathbf{X}^0) = 0.786 > V(\mathbf{X}^*)$, respectively.

4. Numerical experiments

In this section, we perform various numerical experiments to investigate the effect of our volume correction algorithm described in Section 3.2 to overcome the volume loss problem. Unless otherwise specified, we set the density to be $\rho_1 = \rho_2 = 1$ and the volume correction tolerance to be *tol* = 1e−4.

4.1. Pressure jump of drop

In the absence of viscous, gravitational, or other external forces, surface tension forces tend to make a drop spherical. Laplace's formula for a sphere surrounded by a background fluid at pressure $p_{\text{background}}$ gives the internal pressure of the drop p_{drop} to be $p_{\text{drop}} = p_{\text{background}} + 2\sigma/R$, where R is a radius of the drop [1]. In this experiment, a fluid drop is placed at the center of domain $\Omega = (0,1) \times (0,1) \times (0,1)$ with radius $R = 0.2$. The pressure jumps are computed on the uniform grids, $h = 1/2^n$ for $n = 7$, 8, and 9 with corresponding time steps, $\Delta t = 0.5h^2$. The calculations are run up to time $T = \Delta t$. By employing $\mu = 0.1$ and $\sigma = 1$, we have the theoretical pressure jump $[p] = p_{\text{drop}} - p_{\text{background}} = 10$. Table 1 shows a set of computed pressure jumps $[p] = p_{\max} - p_{\min}$. Here $p_{\max}$ and $p_{\min}$ are the maximum and minimum values of the com-

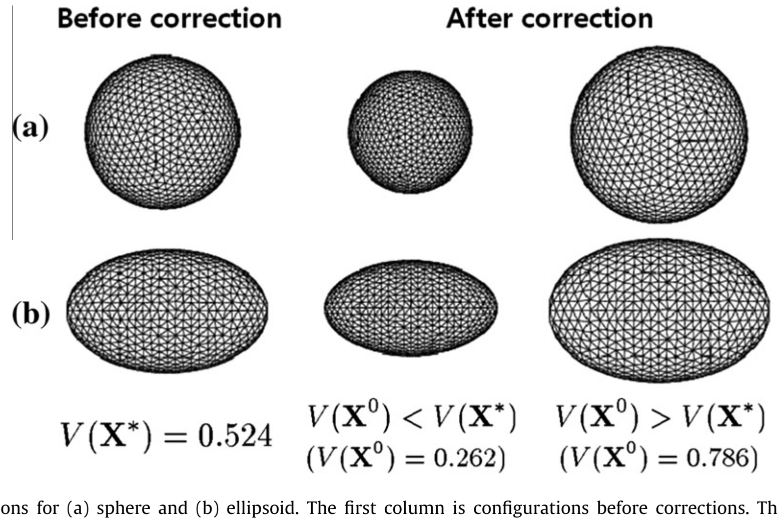
Before correction
After correction
(a)
(b)
$V(\mathbf{X}^*) = 0.524$
$V(\mathbf{X}^0) < V(\mathbf{X}^*)$
$(V(\mathbf{X}^0) = 0.262)$
$V(\mathbf{X}^0) > V(\mathbf{X}^*)$
$(V(\mathbf{X}^0) = 0.786)$

Fig. 9. Examples of volume corrections for (a) sphere and (b) ellipsoid. The first column is configurations before corrections. The second and third columns are after corrections in the cases of $V(\mathbf{X}^0) < V(\mathbf{X}^*)$ and $V(\mathbf{X}^0) > V(\mathbf{X}^*)$, respectively.

Table 1
Pressure jump $[p]$ with $\sigma = 1$ and $R = 0.2$ for different mesh sizes.

Mesh sizes	$32 \times 32 \times 32$	$64 \times 64 \times 64$	$128 \times 128 \times 128$
$[p]$	10.0796	10.0338	10.0326

puted pressure, respectively. The numerical results of the pressure jump are in quantitative agreements with the theoretical value.

4.2. Volume loss by spurious velocities induced by surface tension

We consider a spherical drop in spurious velocities [56]. In incompressible fluids, surface tension forces are balanced by a discontinuity in pressure across the interface. Spurious velocities may result from the inability of discretization schemes to represent this discontinuous pressure. The initial drop has a radius of 0.25 and is at the center of the unit domain. The other parameters are $h = 1/64$, $\Delta t = 0.5h^2$, $\mu = 0.01$, and $\sigma = 50$. In Fig. 10, (a) shows the initial shape, (b and c) depict the shape evolutions at $T = 0.048$ without and with volume corrections, respectively. Without the volume correction, the volume loss is visually evident and is about 17.15% at time T as shown in (d).

4.3. Rotating sphere by background fluid flows

Though a velocity field remains divergence-free on the Lagrangian grid (in a continuous sense), the volume loss can be observed by the discrete time step when the immersed boundary point is updated. To illustrate the comparison between the volume loss and our proposed volume correction method, we consider the passive advection of a sphere by the background velocity field such as

$$\mathbf{u} = (u, v, w) = (8\pi(y - 0.5), -8\pi(x - 0.5), 0). \qquad (33)$$

The sphere with a radius of 0.25 is centered at (0.5,0.5,0.5) in the unit domain as shown in Fig. 11a. We run the computation only with Eq. (6) using the given velocity field, Eq. (33). We use $h = 1/64$ and $\Delta t = 0.5h^2$. Fig. 11b and c shows the shape evolutions of the sphere without and with volume corrections at $T = 0.5$, respectively. Fig. 11d shows the relative volume change with and without corrections. Without correction, the greater volume gains 36.1% from the initial.

Next, we consider another passive advection of a sphere by the background velocity field $\mathbf{u} = (u, v, w) = (8\pi(y - 0.5), 0, -8\pi(x - 0.5))$. The sphere with a radius 0.15 is centered at (0.5,0.5,0.75) in the unit domain. Then with the velocity field, the sphere rotates about the axis $(0.5, y, 0.5)$ with the period of time 0.25. Thus the drop reaches the same position after $T = 0.25$. With the same numerical parameters as those used in the above test, the shape evolutions are shown in Fig. 12a. Figs. 12(b–d) shows the initial shape, the final shape without and with volume corrections, respectively. We can observe that the background velocity results in volume gains without volume correction. It is worth noting here that there is additional error introduced from the interpolation operator, which is related to the discrete delta function.

4.4. Convergence test and computational cost

To calculate the convergence rate, we consider the numerical investigations with the same first test problem set in Section 4.3. The numerical solutions are computed on the uniform grids, $h = 1/2^n$, for $n = 6, 7$, and 8 and with corresponding time steps, $\Delta t = 0.1h^2$. We define the discrete l_2-norm error e_n as $e_n = \sqrt{\frac{1}{M}\sum_{l=1}^{M} |\mathbf{X}_l^T - \mathbf{X}_l^0|^2}$. The rate of convergence is defined as the ratio of successive errors: $\log_2(e_n/e_{n+1})$. Using these definitions, the errors and rates of convergence are given in Table 2. The results

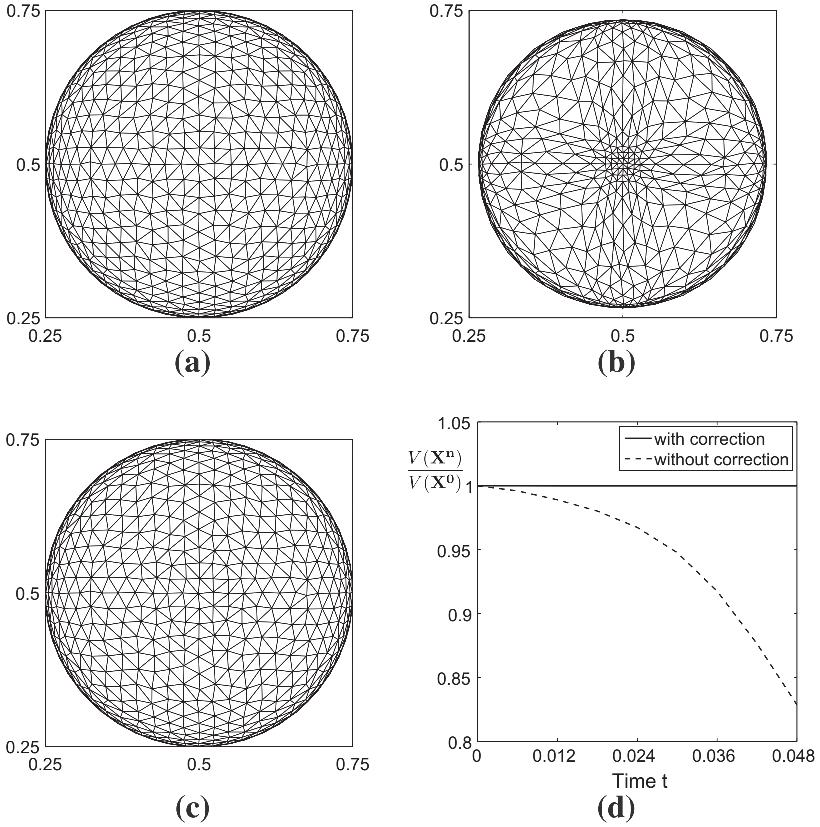

Fig. 10. Volume loss test by spurious velocities induced by surface tension. (a) Initial shape, (b) without volume correction at $T = 0.048$, (c) with volume correction at $T = 0.048$, and (d) relative volume loss $V(\mathbf{X}^n)/V(\mathbf{X}^0)$ with and without corrections.

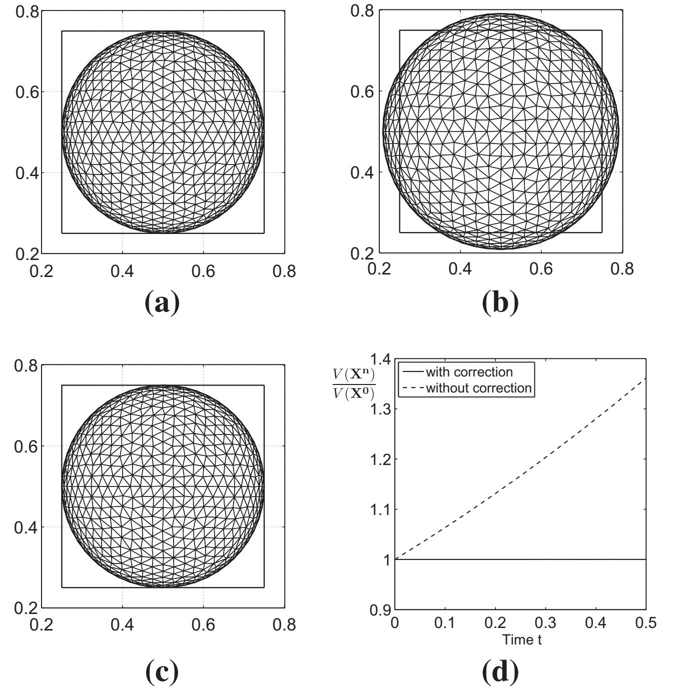

Fig. 11. Rotating sphere by background fluid flows. (a) Initial shape, (b) without volume correction at $T = 0.5$, (c) with volume correction at $T = 0.5$, and (d) relative difference of volume $V(\mathbf{X}^n)/V(\mathbf{X}^0)$ with and without corrections.

suggest that the scheme has indeed the second order accurate in space and the first order accurate in time.

Next, we compute the computational cost with different tolerance values and time steps when the volume correction algorithm is applied. The problem is the same as above except $T = 1/2$, $h = 1/128$, and a time step. Table 3 shows CPU times with different time steps and tolerance values for without and with volume correction algorithms. This result suggests that the additional CPU time due to the proposed volume correction algorithm is negligible. This also means that the rest of the computational time is simply interpolating the velocity and moving the boundary.

Table 2
Error and convergence rates.

Grid size	64^3		128^3		256^3
e_n	2.274e−4		5.781e−5		1.452e−5
Rate		1.98		1.99	

Table 3
CPU times with different time steps and tolerance values. Note that A is the number of volume correction algorithms which is taken from B iterations in (A/B).

Time step	CPU time (s)	CPU time (s)	With correction
Δt	Without correction	*tol* = 1e−4	*tol* = 1e−5
h^2	336.9	338.5 (190/8192)	340.9 (1638/8192)
$0.5h^2$	674.3	674.9 (96/16384)	676.3 (963/16384)

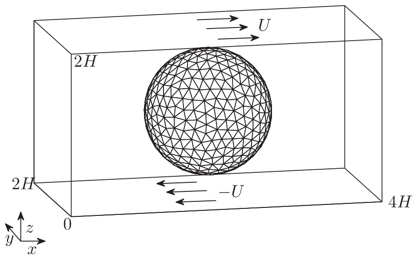

Fig. 13. Schematic illustration of the initial and boundary conditions.

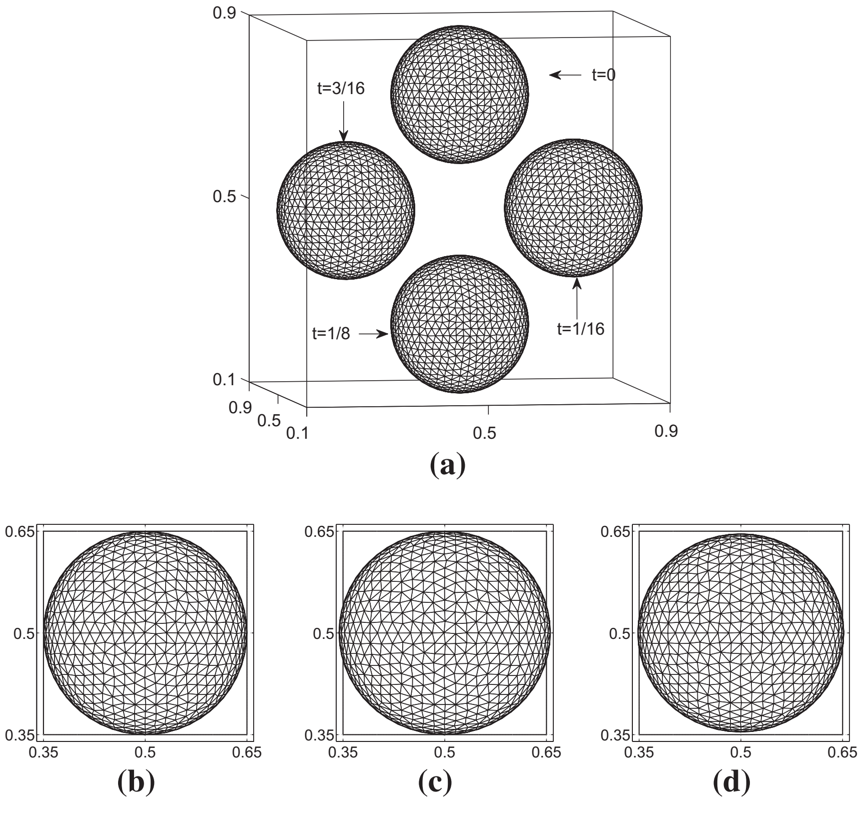

Fig. 12. (a) Evolution of a drop by the background velocity field. (b) Initial shape, (c) without volume correction at $T = 0.25$, and (d) with volume correction at $T = 0.25$.

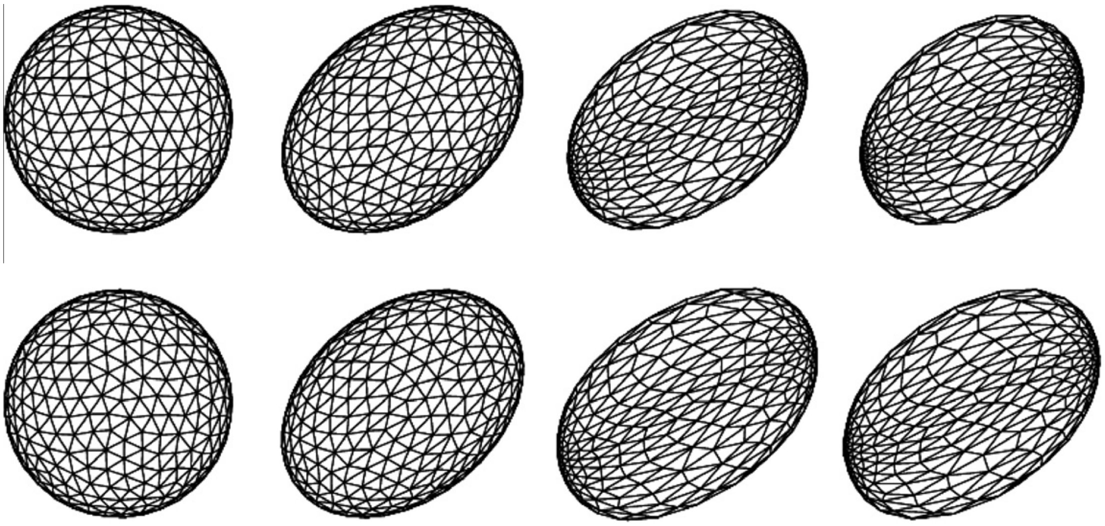

Fig. 14. Evolution of a droplet in the shear flow. Top: Results without volume correction. Bottom: Results with volume correction. From left to right, computational times are $t = 0,\ 0.20.8$, and 1.2.

Furthermore, the NS equations had been solved at every time step, the fraction of time spent performing the volume correction would be even smaller than reported in this test.

4.5. A droplet in shear flows

We investigate the deformation of a droplet in shear flows. The initial droplet is assumed to be a spherical shape with radius R centered at $(2H, H, H)$ in the computational domain $\Omega = (0, 4H) \times (0, 2H) \times (0, 2H)$. The upper plate speed is U and the lower one is $-U$. Fig. 13 shows a schematic illustration of the initial and boundary conditions.

Now we begin with a drop positioned at the center of the computational domain $\Omega = (0, 4) \times (0, 2) \times (0, 2)$. In this simulation we take the parameters: $R = 0.5,\ H = 2,\ U = 1,\ \mu = 0.1$, and $\sigma = 1$. Here, $128 \times 64 \times 64$ grid points and the time step $\Delta t = 0.1h^2$ are used. The calculations are run up to the time $T = 1.2$.

The top and bottom rows of Fig. 14 show the shape evolution of the droplet without and with volume corrections, respectively. We observe that without volume correction the droplet keeps shrinking and cannot reach the steady state due to the volume loss. On the other hand, the drop has reached to the steady state with the volume correction method.

Also to verify the accuracy of the proposed method, we compare it with several previous results. Those are from volume of fluid computations of Li et al. [57], boundary integral computations of Rallison [58], lattice Boltzmann simulations by Inamuro et al. [59], experimental results by Rumscheidt and Mason [60], and theoretical result by Cox [61]. We recall the definition of the drop deformation $D = (L - B)/(L + B)$, where L and B are the maximum and minimum drop dimensions in the xz plane, respectively. Also θ is the angle between the maximum axis of deformation and the x axis. Previous simulations were performed with various capillary numbers $Ca = 0.1, 0.2, 0.3$, and 0.4 with fixed Reynolds number $Re = 0.2$. In our case, we fix $\mu = 5$ and set $\sigma = \mu/Ca$. The initial spherical drop with radius 0.25 is set at the center of the computational domain $\Omega = (0, 2) \times (0, 1) \times (0, 2)$. $64 \times 32 \times 64$ grid points and the time step $\Delta t = 0.01h^2$ are used. Plots of D and θ at steady states are shown in Fig. 15a and b, respectively. To compare the proposed algorithm to the previous results, we put them together. From these results, we observe that the results obtained by our proposed method are in good agreement with the previous numerical methods [57–59], experimental [60], and theoretical results [61].

4.6. Rising bubble

We simulate the rising bubble with a high density ratio $\rho_1/\rho_2 = 1/1000$ with the gravity $\mathbf{g} = (0, 0, -1)$. The computation is performed in a domain $(0, 1) \times (0, 1) \times (0, 2)$ with a mesh size $h = 1/32$. A periodic boundary condition is specified in the x- and y-directions. On the bottom and top sides of the domain,

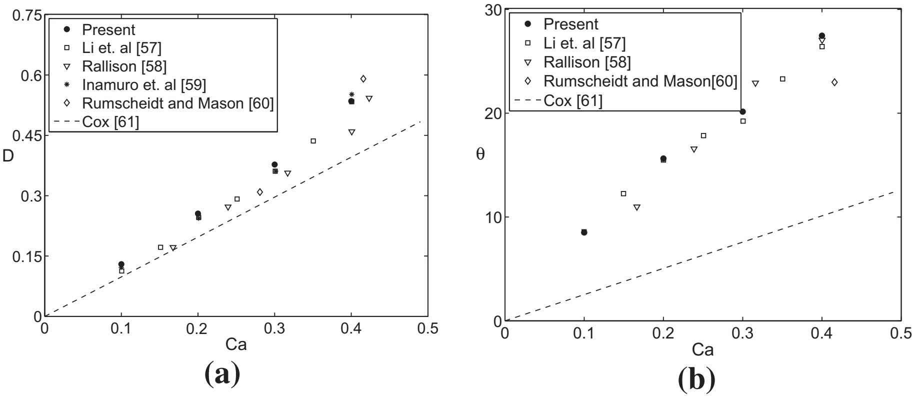

Fig. 15. Comparison between our method and previous methods for (a) drop deformation and (b) drop angle.

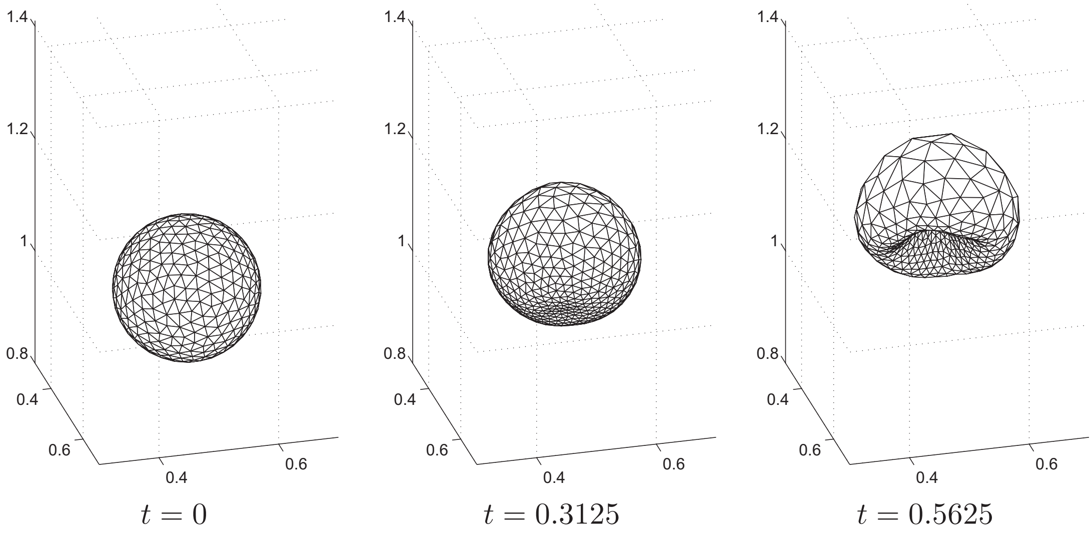

Fig. 16. Shape evolution of a rising bubble at time $t = 0$, 0.3125, and 0.5625.

$p_z = -\rho^n$ and zero velocity are applied, respectively. In addition, the other parameters $\Delta t = 1.6h^2$, $\mu = 0.01$ and $\sigma = 0.1$ are used. Fig. 16 shows the rising bubble, which is initially centered at $(0.5, 0.5, 1)$ with a radius of 0.12. The bubble rises and deforms due to the buoyancy-driven force.

5. Conclusions

We have presented the volume-preserving scheme for two-phase immiscible incompressible flows using the IBM in a three-dimensional space. The present study extended the previous two-dimensional research [35] to the three-dimensional space. The key idea of the proposed method is relocating surface points along the normal directions to preserve the total volume. We performed various numerical experiments to show the efficiency and accuracy of the proposed method. Moreover, additional CPU time for the volume correction algorithm was negligible. The results demonstrated that the volume-preserving algorithm has the potential to solve many applications using the volume constrained immersed boundary method.

Acknowledgments

J.S. Kim was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2011-0023794). The authors thank Ha-Kyu Song for many useful discussions. The authors also wish to thank the reviewers for the constructive and helpful comments on the revision of this article.

References

[1] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335–354.
[2] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, W.J. Rider, A high-order projection method for tracking fluid interfaces in variable density incompressible flows, J. Comput. Phys. 130 (1997) 269–282.
[3] J.E. Pilliod, E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, J. Comput. Phys. 199 (2004) 465–502.
[4] S. Osher, R.P. Fedkiw, Level set methods: an overview and some recent results, J. Comput. Phys. 169 (2001) 463–502.
[5] M. Sussman, P. Smereka, S. Osher, A level set approach for incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[6] Y.C. Chang, T.Y. Hou, B. Merriman, S. Osher, A level set formulation of Eulerian interface capturing methods for incompressible fluid flows, J. Comput. Phys. 124 (1996) 449–464.
[7] E. Olsson, G. Kreiss, A conservative level set method for two phase flow, J. Comput. Phys. 210 (2005) 225–246.
[8] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708–759.
[9] O.S. Galaktionov, P.D. Anderson, G.W.M. Peters, F.N. Van de Vosse, An adaptive front tracking technique for three-dimensional transient flows, Int. J. Numer. Meth. Fluids 32 (2000) 201–217.
[10] S.O. Unverdi, G. Trayggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, J. Comput. Phys. 100 (1992) 25–37.
[11] D. Jamet, O. Lebaigue, N. Coutris, J.M. Delhaye, The second gradient method for the direct numerical simulation of liquid–vapor flows with phase change, J. Comput. Phys. 169 (2001) 624–651.
[12] J.S. Kim, A continuous surface tension force formulation for diffuse-interface models, J. Comput. Phys. 204 (2005) 784–804.
[13] J.S. Kim, A generalized continuous surface tension force formulation for phase-field models for multi-component immiscible fluid flows, Comput. Methods Appl. Mech. Eng. 198 (2009) 3105–3112.
[14] S. Chen, G.D. Doolen, Lattice Boltzmann method for fluid flows, Annu. Rev. Fluid Mech. 30 (1998) 329–364.
[15] T. Inamuro, T. Ogata, S. Tajima, N. Konishi, A lattice Boltzmann method for incompressible two-phase flows with large density differences, J. Comput. Phys. 198 (2004) 628–644.
[16] C. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[17] R.J. Leveque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.
[18] D. Boffi, L. Gastaldi, L. Heltai, S. Peskin, On the hyper-elastic formulation of the immersed boundary method, Comput. Methods Appl. Mech. Eng. 197 (2008) 2210–2231.
[19] W.-X. Huang, C.B. Chang, H.J. Sung, Three-dimensional simulation of elastic capsules in shear flow by the penalty immersed boundary method, J. Comput. Phys. 231 (2012) 3340–3364.
[20] W.-X. Huang, C.B. Chang, H.J. Sung, An improved penalty immersed boundary method for fluid–flexible body interaction, J. Comput. Phys. 230 (2011) 5061–5079.
[21] Y. Kim, C. Peskin, Numerical study of incompressible fluid dynamics with nonuniform density by the immersed boundary method, Phys. Fluids 20 (2008) 06210101.
[22] M. Francois, W. Shyy, Computations of drop dynamics with the immersed boundary method, Part 1: Numerical algorithm and buoyancy-induced effect, Numer. Heat Tranfer Part B 44 (2003) 101–118.
[23] M. Francois, W. Shyy, Computations of drop dynamics with the immersed boundary method, Part 2: Drop impact and heat transfer, Numer. Heat Tranfer Part B 44 (2003) 119–143.
[24] M. Francois, E. Uzgoren, J. Jackson, W. Shyy, Multigrid computations with the immersed boundary technique for multiphase flows, Int. J. Numer. Methods H 14 (2004) 98–115.
[25] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35–60.
[26] Y. Tseng, J.H. Ferziger, A ghost-cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2003) 593–623.
[27] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509–534.

[28] L. Zhu, R.C.Y. Chin, Simulation of elastic filaments interacting with a viscous pulsatile flow, Comput. Methods Appl. Mech. Eng. 197 (2008) 2265–2274.
[29] L. Zhu, C. Peskin, Simulation of a flexible flapping filament in a flowing soap film by the immersed boundary method, J. Comput. Phys. 179 (2002) 452–468.
[30] M.-C. Lai, C. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.
[31] S. Xu, Z.J. Wang, An immersed interface method for simulating the interaction of a fluid with moving boundaries, J. Comput. Phys. 216 (2006) 454–493.
[32] C. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.
[33] E. Newren, A. Fogelson, R. Guy, R. Kirby, Unconditionally stable discretizations of the immersed boundary equations, J. Comput. Phys. 222 (2007) 702–719.
[34] C. Peskin, B. Printz, Improved volume conservation in the computation of flows with immersed elastic boundaries, J. Comput. Phys. 105 (1993) 33–46.
[35] Y. Li, E. Jung, W. Lee, H.G. Lee, J.S. Kim, Volume preserving immersed boundary methods for two-phase fluid flows, Int. J. Numer. Methods Fluids 69 (2012) 842–858.
[36] D. Boffi, N. Cavallini, F. Gardini, L. Gastaldi, Immersed boundary method: performance analysis of popular finite element spaces, Coupled Prob. 2011 (2011) 1–12.
[37] D. Boffi, N. Cavallini, F. Gardini, L. Gastaldi, Local mass conservation of stokes finite elements, J. Sci. Comput. 52 (2012) 383–400.
[38] P.O. Persson, G. Stangt, A simple mesh generator in MATLAB, SIAM Rev. 46 (2004) 329–345.
[39] P.O. Persson, Mesh size functions for implicit geometries and PDE-based gradient limiting, Eng. Comput. 22 (2006) 95–109.
[40] C.B. Barber, D.P. Dobkin, H.T. Huhdanpaa, The quickhull algorithm for convex hulls, TOMS 22 (1996) 469–483.
[41] National Science and Technology Research Center for Computation and Visualization of Geometric Structures, The Geometry Center, University of Minnesota, 1993.
[42] F. Harlow, J. Welch, Numerical calculation of time dependent viscous incompressible flow with free surface, Phys. Fluids 8 (1965) 2182–2189.
[43] M. Meyer, M. Desbrun, P. Schröder, A.H. Barr, Discrete differential-geometry operators for triangulated 2-manifolds, in: Proc. VisMath., 2002, pp. 35–57.
[44] G. Xu, Discrete Laplace–Beltrami operator on sphere and optimal spherical triangulations, Int. J. Comput. Geom. Appl. 16 (2006) 75–93.
[45] M. Belkin, J. Sun, Y. Wang, Discrete laplace operator on meshed surfaces, Proc. SCG (2008) 278–287.
[46] M. Wardetzky, Convergence of the cotangent formula: an overview, Discr. Differ. Geom. 38 (2008) 275–286.
[47] J.M. Sullivan, Curvatures of smooth and discrete surfaces, Discr. Differ. Geom. 38 (2008) 175–188.
[48] C.L. Bajaj, G. Xu, Anisotropic diffusion of surfaces and functions on surfaces, J. ACM Trans. Graph. 22 (2003) 4–32.
[49] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (104) (1968) 745–762.
[50] U. Trottenberg, C. Oosterlee, A. Schüller, Multigrid, Academic press, London, 2001.
[51] A.V. Gelder, Efficient computation of polygon area and polyhedron volume, in: A.W. Paeth (Ed.), Graphics Gems V, Academic Press, New York, 1995.
[52] R. Goldman, Area of planar polygons and volume of polyhedra, in: J. Arvo (Ed.), Graphics Gems II, Academic Press, New York, 1994.
[53] M.J. Aftosmis, M.J. Berger, J.E. Melton, Robust and efficient cartesian mesh generation for component-based geometry, AIAA J. 36 (1998) 952–960.
[54] N.H. Trinh, J. Lester, B.C. Fleming, G. Tung, B.B. Kimia, Accurate measurement of cartilage morphology using a 3D laser scanner, CVAMIA 4241 (2006) 37–48.
[55] B.L. Lind, J. Fagertun, J.E. Wilhjelm, M.S. Jensen, H. Sillesen, 3D reconstruction of carotid atherosclerotic plaque: comparison between spatial compound ultrasound models and anatomical models, Ultras. Med. Biol. 33 (2007) 1064–1074.
[56] S. Ganesan, G. Matthies, L. Tobiska, On spurious velocities in incompressible flow problems with interfaces, Comput. Methods Appl. Mech. Eng. 196 (2007) 1193–1202.
[57] J. Li, Y.Y. Renardy, M. Renardy, Numerical simulation of breakup of a viscous drop in simple shear flow through a volume-of-fluid method, Phys. Fluids 12 (2000) 269–282.
[58] J.M. Rallison, The deformation of small viscous drops and bubbles in shear flows, Annu. Rev. Fluid Mech. 16 (1984) 45–66.
[59] T. Inamuro, R. Tomita, F. Ogino, Lattice Boltzmann simulations of drop deformation and breakup in shear flows, Int. J. Mod. Phys. B 17 (2003) 21–26.
[60] F.D. Rumscheidt, S.G. Mason, Particle motions in sheared suspensions XII. Deformation and burst of fluid drops in shear and hyperbolic flow, J. Colloid Sci. 16 (1961) 238–261.
[61] R.G. Cox, The deformation of a drop in a general time-dependent fluid flow, J. Fluid Mech. 37 (1969) 601–623.