COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. **3**, No. 3, pp. 582-602

Commun. Comput. Phys.
March 2008

A General Moving Mesh Framework in 3D and its Application for Simulating the Mixture of Multi-Phase Flows

Yana Di[1], Ruo Li[2,*] and Tao Tang[3,4]

[1] *P.O. Box 2719, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100080, China.*
[2] *LMAM & School of Mathematical Sciences, Peking University, Beijing 100871, China.*
[3] *Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Kowloon, Hong Kong.*
[4] *Faculty of Science, Beijing University of Aeronautics and Astronautics, Beijing 100083, China.*

Received 20 December 2006; Accepted (in revised version) 10 June 2007

Communicated by Jie Shen

Available online 30 October 2007

Abstract. In this paper, we present an adaptive moving mesh algorithm for meshes of unstructured polyhedra in three space dimensions. The algorithm automatically adjusts the size of the elements with time and position in the physical domain to resolve the relevant scales in multiscale physical systems while minimizing computational costs. The algorithm is a generalization of the moving mesh methods based on harmonic mappings developed by Li et al. [J. Comput. Phys., 170 (2001), pp. 562-588, and 177 (2002), pp. 365-393]. To make 3D moving mesh simulations possible, the key is to develop an efficient mesh redistribution procedure so that this part will cost as little as possible comparing with the solution evolution part. Since the mesh redistribution procedure normally requires to solve large size matrix equations, we will describe a procedure to decouple the matrix equation to a much simpler block-tridiagonal type which can be efficiently solved by a particularly designed multi-grid method. To demonstrate the performance of the proposed 3D moving mesh strategy, the algorithm is implemented in finite element simulations of fluid-fluid interface interactions in multiphase flows. To demonstrate the main ideas, we consider the formation of drops by using an energetic variational phase field model which describes the motion of mixtures of two incompressible fluids. Numerical results on two- and three-dimensional simulations will be presented.

AMS subject classifications: 65M20, 65M50, 65M60

Key words: Moving mesh methods, multi-phase flows, unstructured tetrahedra, phase field model, Navier-Stokes equations, finite element method.

*Corresponding author. *Email addresses:* yndi@lsec.cc.ac.cn (Y.-N. Di), rli@math.pku.edu.cn (R. Li), ttang@math.hkbu.edu.hk (T. Tang)

1 Introduction

Several moving mesh techniques have been introduced in the past, in which the most advocated method is the one based on solving elliptic PDEs first proposed by Winslow [31]. Winslow's formulation requires the solution of a nonlinear, Poisson-like equation to generate a mapping from a regular domain in a parameter space Ω_c to an irregularly shaped domain in physical space Ω. Brackbill and Saltzman [5] formulated the grid equations in a variational form to produce satisfactory mesh concentration while maintaining relatively good smoothness and orthogonality. Their approach has become one of the most popular methods used for mesh generation and adaptation. Dvinsky [15] suggests the possibility that harmonic function theory may provide a general framework for developing useful mesh generators. His method can be viewed as a generalization and extension of Winslow's method. However, unlike most other generalizations which add terms or functionals to the basic Winslow grid generator, his approach uses a single functional to accomplish the adaptive mapping. The critical points of this functional are *harmonic maps*. Meshes obtained by Dvinsky's method enjoy desirable properties of harmonic maps, particularly regularity and smoothness.

Motivated by the work of Dvinsky, a moving mesh finite element strategy based on harmonic mapping was proposed and studied by Li et al. in [19]. The key idea is to construct the harmonic map between the physical space and a parameter space by an iterative procedure. In [20], a moving mesh method based on the minimization of the mesh energy is proposed which seems having big potential for simulations in high space dimensions. More precisely, in the mesh points re-distribution step, we solve an optimization problem with some appropriate constraints, which is in contrast to the traditional method of solving the Euler-Lagrange equation directly. The key idea of this approach is to treat the interior and boundary grids as a whole, rather than considering them separately.

In this paper, we present an adaptive moving mesh algorithm for meshes of unstructured tetrahedra in three dimensions, which is a generalization of the moving mesh methods based on harmonic mappings developed by Li et al. in [19, 20]. To make 3D moving mesh simulations possible, the key is to provide a more efficient mesh redistribution procedure so that this part will cost very little comparing with the solution evolution part. Since the mesh re-distribution procedure normally requires to solve large scaled algebraic systems (arising from discretizing the Euler-Lagrange equations or the minimization problem), we will describe a procedure to decouple the matrix equation to a much simpler block-tridiagonal type which can be solved by multi-grid solvers very efficiently. The proposed algebraic multigrid solver deals with the nonlinear constraints locally in the smoothing operation, so that the multigrid procedure is comparative to some pointwise Gauss-Seidel iterations. This allows us to avoid solving a saddle point problem.

To demonstrate the performance of the proposed 3D moving mesh strategy, the algorithm is implemented in finite element simulations of deformable droplet and fluid-fluid interface interactions for multi-phase flows. The flow we consider has discontinuous

density and viscosity, and is characterized by large density and viscosity ratios at the free surface, e.g. air and water. The main existing computational methods used to solve incompressible two-phase flow problems include front-tracking methods, boundary integral methods, volume-of-fluid methods, and level set methods, see, e.g., [22,26,30]. Alternative sharp interface works include boundary condition capturing methods [18] and finite element based level set methods [13,29] for simulating multiphase incompressible flows. In this work, the governing equations describing the motion of mixtures of two incompressible fluids are based on an energetic variational phase field models studied recently by Liu, Shen et al. [21,34]. As demonstrated in [34], the diffuse-interface model will stay close to the sharp-interface model, with the conventional interfacial tension when the interfacial thickness tends to zero. Note that the phase function ϕ profile is "nontrivial" only within the interfacial layer. The interfacial profile needs to be numerically resolved for accurate evaluation of the interfacial stress, and the disparity between small thickness and the global length scale implies the need for a locally refined grid inside the interfacial region.

Recently, methods that couple two different schemes have been investigated for simulating fluid flows with moving interfaces. Examples are the coupled level set and volume-of-fluid (VOF) method [25], the hybrid particle level set method [16], and the mixed markers and VOF method [2]. A coupled method takes advantage of the strengths of each of the two methods, and are therefore superior to either method alone. More recently, Yang et al. [33] proposed an adaptive coupled level set and VOF volume tracking method for unstructured triangular grids. The use of the adaptive unstructured grids can cluster the grid near the interface, and therefore enhance the efficiency and accuracy for solving the interface structures. The adaptive coupled level set and VOF volume tracking technique has been demonstrated powerful in resolving complex interface changes and interfaces of high curvature. The spatially adaptive techniques for level set methods and incompressible flow have been recently reviewed by Losasso et al. [23] who discussed both historical and most recent works in this research direction.

In the present work, we will apply our 3D moving mesh algorithm to simulate fluid flows with moving interfaces. Ideally, for the moving interface problems we would like the mesh to be clustered within the interface region to correctly capture the effects of surface tension, while away from the interface sufficient resolution may be obtained using less grid points. It will be demonstrated that the moving-mesh-phase-field approach can resolve complex interface structures very efficiently. The proposed method is relatively simple and, compared to non-adaptive method, requires fewer elements while still keeping the mesh sufficiently refined near the interfaces. The mesh quality near the interface will be also examined.

The rest of the paper is organized as follows. In Section 2, we describe a general moving mesh strategy which is independent of the space dimensions. The multigrid method is employed to speed up the mesh redistribution procedure. In Section 3, we briefly describe a phase field model for two incompressible fluids, and a finite element discretization for the model. In Section 4, a number of numerical examples are consid-

ered. It is demonstrated that the general moving mesh strategy proposed in this work is very effective in simulating the mixture of multi-phase flows. The final section contains some concluding remarks.

2 Moving mesh strategy

Let Ω and Ω_c be compact Riemannian manifolds of dimension n with metric tensors d_{ij} and $r_{\alpha\beta}$ in some local coordinates $\vec{x}$ and $\vec{\xi}$, respectively. It has been demonstrated (see, e.g., [6]) that the Euler-Lagrange equations, with Euclidean metric for the logical domain Ω_c, are given by

$$\frac{\partial}{\partial x^i}\left(G^{ij}\frac{\partial \xi^k}{\partial x^j}\right)=0, \tag{2.1}$$

and the corresponding mesh energy is of the form

$$E(\vec{\xi})=\sum_k \int_\Omega G^{ij}\frac{\partial \xi^k}{\partial x^i}\frac{\partial \xi^k}{\partial x^j}d\vec{x}, \tag{2.2}$$

where the inverse of (G^{ij}) is called the *monitor function*. Solutions to (2.1) are harmonic functions giving a continuous and one-to-one mapping which is differentiable and has a non-zero Jacobian.

In Li et al. [20], an effective approach which re-distributes the interior and boundary grids *simultaneously* was introduced. The method uses an optimization approach to generate the adaptive grids, which turns out to be dimension-independent. In this section, we will improve the algorithm in [20] so that it is useful for solving large scale simulations in three space dimensions. Since the mesh redistribution procedure as described below normally requires to solve large-size matrix equations arising from solving the minimization problems, we will describe a procedure to decouple the matrix equation to a much simpler block-tridiagonal type which can be solved very efficiently by multi-grid methods.

2.1 Mesh-redistribution algorithm

Here we briefly outline the main idea of the optimization-based grid redistribution approach based on Algorithm 2.1.

More details can be found in [19,20] for a 2D solution domain and [11,14] for a spherical domain.

2.2 Mesh re-distribution and speeding up the procedure

Let us discretize the optimization problem (2.3) in the linear finite element space. The triangulation of the physical domain is $\mathcal{T}$, with T_i as its elements, and X_i as its nodes. The

Algorithm 2.1: Mesh-redistribution algorithm

1. Solve the optimization problem

$$\min \sum_k \int_\Omega G^{ij} \frac{\partial \xi^k}{\partial x^i} \frac{\partial \xi^k}{\partial x^j} d\vec{x} \qquad \text{s.t.} \quad \xi|_{\partial\Omega} = \xi_b \in \mathbf{K}, \tag{2.3}$$

   and compute the L^∞-difference between the solution and the fixed (initial) mesh in the logical domain. If the difference is smaller than a preassigned tolerance, then the mesh-redistribution at the given time level is complete. Otherwise, do

2. Obtain the direction and the magnitude of the movement for $\vec{x}$ by using the difference obtained in Step 1, and then move the mesh;
3. Update the physical solution $\vec{u}$ on the new grid by solving a system of continuous equations;
4. Update the monitor function by using $\vec{u}$ obtained in Step 3, and go to Step 1.

corresponding triangulation on the computational domain is $\mathcal{T}_c$, with $T_{i,c}$ as its elements, and Ξ_i as its nodes. The linear finite element space on the mesh is denoted by $H_h^1(\Omega)$. If the basis function on the node X_i is denoted by ϕ^i, then ξ can be approximated by $\xi_i \phi^i$ (here the standard summation convention is assumed). The coordinates of X_i are $(X_i^1 \ X_i^2 \ X_i^3)^T$. Let the inner nodes be indexed from 1 to N_{inner} and the boundary nodes be indexed from $N_{inner}+1$ to N. The coordinates of the nodes Ξ_i in the computational domain are denoted by $(\Xi_i^1 \ \Xi_i^2 \ \Xi_i^3)^T$. Denote $X = (X^1 \ X^2 \ X^3)^T$, $\Xi = (\Xi^1 \ \Xi^2 \ \Xi^3)^T$, where $X^k = (X_1^k \cdots X_N^k)^T$, $\Xi^k = (\Xi_1^k \cdots \Xi_N^k)^T$, $k = 1,2,3$.

We now discuss how to approximate (2.3). First a linear system for Ξ will be formed to determine the motion of the computational grids. Denote

$$H = \left(\int_\Omega G^{ij} \frac{\partial \phi^\alpha}{\partial x^i} \frac{\partial \phi^\beta}{\partial x^j} d\vec{x} \right)_{1 \le \alpha, \beta \le N.} \tag{2.4}$$

We further split the matrix H into the following form:

$$H = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix} \qquad \begin{matrix} \longleftarrow 1 \text{ to } N_{inner} \text{ row} \\ \longleftarrow N_{inner}+1 \text{ to } N \text{ row} \end{matrix}$$

$$\begin{matrix} \uparrow & \uparrow \\ 1 \text{ to } N_{inner} \text{ column} & N_{inner}+1 \text{ to } N \text{ column} \end{matrix}$$

Assume that the constraint leads to a linear system of the form

$$\sum_k A_k \Xi^k = b,$$

or equivalently,

$$\sum_k A_{k,inner} \Xi_{inner}^k + \sum_k A_{k,bound} \Xi_{bound}^k = b.$$

Observe that $A_{k,inner}=0$ and the matrices $A_{k,bound}$ are the entries of the unit normal of the boundary segments. Then the optimization problem (2.3) is equivalent to the following linear constrained optimization problem:

$$\min_{\Xi}\sum_k\left\{\Xi^{k,T}H\Xi^k\right\} \qquad \text{s.t.} \quad \sum_k A_k\Xi^k=b. \tag{2.5}$$

In fact, (2.5) leads to a linear system with a Lagrangian multiplier λ. In [20], it is proposed to decouple the system (2.5) to a smaller form

$$\begin{pmatrix} H_{22} & 0 & 0 & A_{1,bound}^T \\ 0 & H_{22} & 0 & A_{2,bound}^T \\ 0 & 0 & H_{22} & A_{3,bound}^T \\ A_{1,bound} & A_{2,bound} & A_{3,bound} & 0 \end{pmatrix} \begin{pmatrix} \Xi_{bound}^1 \\ \Xi_{bound}^2 \\ \Xi_{bound}^3 \\ \lambda \end{pmatrix} = \begin{pmatrix} -H_{21}\Xi_{inner}^1 \\ -H_{21}\Xi_{inner}^2 \\ -H_{21}\Xi_{inner}^3 \\ b \end{pmatrix}, \tag{2.6}$$

which contains boundary cells only, together with a symmetric and positive definite system

$$\begin{pmatrix} H_{11} & 0 & 0 \\ 0 & H_{11} & 0 \\ 0 & 0 & H_{11} \end{pmatrix} \begin{pmatrix} \Xi_{inner}^1 \\ \Xi_{inner}^2 \\ \Xi_{inner}^3 \end{pmatrix} = \begin{pmatrix} -H_{12}\Xi_{bound}^1 \\ -H_{12}\Xi_{bound}^2 \\ -H_{12}\Xi_{bound}^3 \end{pmatrix} a, \tag{2.7}$$

which is solved by using the multigrid method. However, for three-dimensional problems, the small system (2.6) becomes a two-dimensional problem and the resulting algebraic systems may contain thousands of elements. Since (2.6) is not a positive definite system, it is still expensive to solve if the matrix size is large. A detailed description of the above optimization procedure can be found in [20].

2.3 Multi-level speed up for (2.7) with a constraint

Since the solution of the optimization system is for the location of mesh grids coordinates, instead of the physical solution, less accurate but efficient algorithms can be employed. In the following we will describe a special procedure to solve the minimization problem (2.3) efficiently. The algebraic multi-grid technique [7] is used to construct the framework of such an algorithm. The main difficult is the constraint on the boundary. It is natural to require that Ξ_{bound}^k is always on the corresponding boundary during the mesh redistribution procedure. This can be guaranteed by requiring that

$$\sum_k A_{k,bound}\Xi_{bound}^k=b. \tag{2.8}$$

It is noted that $A_{k,bound}$ can be some operator and may be extended to the nonlinear case.

Here we describe an algorithm (Algorithm 2.2) for solving $H\Xi^k=0$ with the constraint (2.8) using a multi-level iteration technique.

The procedure performs the n-level algorithm with v_1 pre- and v_2 post-smoothing iterations.

Algorithm 2.2: Multi-level speed up algorithm

- Initialization
  Construct the coarsening matrix H_l, projection matrix P_l, $l=0,\cdots,n$, where n is the level number of the multi-level iteration.
- Multi-level iteration
  1. coarsening process, $l=0,\cdots,n-1$
     - pre-smoothing step: $\Xi_l := S_l^{(v_1)}(\Xi_l, f_l)$
     - restriction step: $f_{l+1} := P_l(H_l\Xi_l - f_l)$
     - $\Xi_{l+1}=0$
  2. smooth at the final level: $\Xi_n = S_n^{(v_1)}(\Xi_n, f_n)$
  3. projection process $l=n-1,\cdots,0$.
     - correction step: $\Xi_l := \Xi_l + P_l^T\Xi_{l+1}$
     - post-smoothing step: $\Xi_l := S_l^{(v_2)}(\Xi_l, f_l)$.

Suppose that a coarse set of points that forms a subset of the fine degrees of freedom (DOFs) has been chosen (strategies for which are discussed below). Then, the fine-level DOFs can be represented as

$$C_0=\{1,2,\cdots,N\}, \quad C_{l-1}=C_l\cup F_l,$$

where C_l is the set of coarse-level points and F_l is the set of remaining fine-level points (so $C_l\cap F_l=\emptyset$). A simple option is chosen for the coarse-grid points where the number of strongly connected neighbors of each point is considered.

Due to the constraints on the boundary, the strategy to setup the coarse-level points should guarantee that in the multigrid iteration procedure, the coarse grid corrections, when applied by the prolongation operator, should not break the boundary constraints on the fine grid. To this end, the grid points in the fine mesh are chosen to be the points of the coarse mesh, with sorted priority based on the constraints applied on it. The points with top-priority are the ones coming from the vertices of the polyhedron, which is the problem domain, followed by the points on the edges, then by the points on the faces, and the interior points are at the lowest priority. To describe the relation between the coarse-grid and the fine-grid, we introduce the "neighborhood" notation B_{li} including the points strongly influenced by the point Ξ_i and

$$B_i\cap B_j=\emptyset \text{ if } i\neq j, \quad \cup_{i=0}^{N_l}B_i=C_{l-1},$$

where N_l is the number of the set C_l.

The smoothing operation is denoted by $S_i(\Xi_i, f_i)$. The most convenient and often effective smoothing process is the Gauss-Seidel iteration, which is also adopted by us.

The decomposition of H_i is introduced as

$$H_i = D_i - L_i - U_i, \tag{2.9}$$

where D_i is the diagonal matrix, L_i (or U_i) is the strictly lower (or upper) triangular matrix. Then one step of the Gauss-Seidel iteration is equivalent to

$$S_i(\Xi_i, f_i) := (D_i - L_i)^{-1}(U_i \Xi_i + f_i). \tag{2.10}$$

Because of the nonlinear constraints, some special treatment for the boundary has to be proposed:

$$\Xi_{bound} := \Xi_{bound} - \delta\Xi_{bound} \cdot \mathbf{n}_l, \tag{2.11}$$

where $\mathbf{n}_l$ is the unit normal of the face Γ_l and $\delta\Xi_{bound}$ is the update introduced by the Gauss-Seidel iteration. Thus the boundary constrain is preserved in the procedure of the Gauss-Seidel iteration, while the Gauss-Seidel iteration is revised into a nonlinear one.

The trivial weighted restriction is chosen as the projection matrix $P_l(N_{l+1} \times N_l)$, i.e., the arithmetic mean average of the neighborhood of each coarse-grid point:

$$P_l(1 \le i \le N_{l+1}, 1 \le j \le N_l) := \begin{cases} \frac{1}{\sharp\{B_{l(C_l)_i}\}} & \text{if } j \in B_{l(C_l)_i}, \\ 0 & \text{otherwise,} \end{cases} \tag{2.12}$$

where $\sharp B$ represents the number of the set B and $(C_l)_i$ represents the ith number of the set C_l. Once a set of coarse points C_l and the projection operator P_l have been chosen, it is natural to define a reasonable prolongation operator and the coarse-level operator by the so-called Galerkin conditions:

$$P_l^T \quad \text{and} \quad H_{l+1} = P_l H_l P_l^T, \ (H_0 = H). \tag{2.13}$$

This way to define the projection operator is very cheap, which depends on the connection of the mesh grids only and ignores the information from the values of the entries in the sparse matrices of the algebraic system. Thus in the total computation, we only need to construct the projection matrix one time.

In this procedure, we treat the interior and boundary grids simultaneously inside the iteration of the multi-level solver. The CPU time consumed in this solver is similar to that for solving equation (2.7). The only difference between them is that in the new solver a boundary projection is added to each smoothing step, i.e., each Gauss-Seidel iteration. But the additional operations (2.11) on the boundary points are local operations which are implemented pointwisely. As a result, the cost is quite low, even if the constraints are extended to the nonlinear case. In general, the cost of one iterative step in the new solver is about the same as that of the standard multigrid iterative step on a $3N \times 3N$ system. Moreover, we need only a few such iterative steps at a given time level since the accuracy requirement of the mesh equations is quite low. Thus the multi-level iteration has accomplished much more efficiency for the new solver compared with the old one in [20]. At the same time, the new solver can handle the nonlinear boundary constrains (2.8) directly, which enables us to re-distribute the meshes along a curved surface.

2.4 Solution updating and monitor function

Since the computational mesh is adapted in time, it is necessary to interpolate all the quantities from the old mesh to the new mesh. Based on some simple observations or assumptions on the numerical solutions, we are able to obtain some continuous interpolation scheme which is to solve some linear PDEs. The PDEs are normally relevant to the governing equations. For example, in case that the governing equations are the incompressible Navier-Stokes equations, the PDEs for the continuous interpolation are the Stokes-like equations which is used in this paper (see [12] for details).

A monitor function is used to guide the mesh redistribution. It may depend on the solution arc-length (in 1D), curvature, gradients and a posteriori errors, see, e.g., [8,9,27, 35]. In this work, the monitor function is chosen as

$$m(\mathbf{x},t)=\sqrt{1+\alpha|\nabla\phi|^2}, \tag{2.14}$$

where α is generally a problem-dependent positive parameter, which is taken as 5 in our numerical experiments. The initial condition for the velocity $\mathbf{u}$ is taken to be zero in all computations.

3 A phase field model for two incompressible fluids

The phase field method (diffuse interface method) as applied to two-phase flows has been described previously by a number of authors [1]. A phase-field function ϕ is introduced such that the concentrations of the two components are $(1+\phi)/2$ and $(1-\phi)/2$, respectively. For the elastic (mixing) energy, we adopt the familiar Ginzburg-Landau form:

$$W(\phi,\nabla\phi)=\int_\Omega\left\{\frac{1}{2}|\nabla\phi|^2+F(\phi)\right\}dx, \tag{3.1}$$

where

$$F(\phi)=(|\phi|^2-1)^2/4\eta^2$$

is the usual double-well form of the bulk energy with η as the capillary width (width of the mixing layer). The evolution of ϕ is governed by the Allen-Cahn equation:

$$\phi_t=-\left(\gamma\nabla\frac{\delta W}{\delta\phi}\right)=\gamma(\Delta\phi-f(\phi)), \tag{3.2}$$

where $\delta W/\delta\phi$ represents the variation of the energy W with respect to ϕ, $f(\phi)$ is a polynomial of ϕ such that $f(\phi)=F'(\phi)$ and γ represents the elastic relaxation time of the system.

A variational procedure applied to the free energy will yield the elastic stress tensor for the system:

$$\mathbf{B}=-\frac{\delta W}{\delta\mathbf{x}}=-\nabla\phi\otimes\nabla\phi-f(\phi)\nabla\phi, \tag{3.3}$$

where

$$(\nabla\phi\otimes\nabla\phi)_{ij}=\nabla_i\phi\nabla_j\phi.$$

The viscous stress tensor is obtained by,

$$\tau=\mu[\nabla\mathbf{u}+(\nabla\mathbf{u})^T]=\left(\frac{1-\phi}{2}\mu_1+\frac{1+\phi}{2}\mu_2\right)[\nabla\mathbf{u}+(\nabla\mathbf{u})^T], \tag{3.4}$$

where μ_1 and μ_2 are the viscosity constants for two components.

Adding the momentum equation, we obtain the following system modeling a specific type of mixture of two incompressible fluids [21]:

$$\rho(\mathbf{u}_t+\mathbf{u}\cdot\nabla\mathbf{u})=\nabla\cdot(-p\mathbf{I}+\tau)-\lambda\nabla\cdot(\nabla\phi\otimes\nabla\phi)+\rho\mathbf{g}(x), \tag{3.5a}$$

$$\nabla\cdot(\rho\mathbf{u})=0, \tag{3.5b}$$

$$\phi_t+\mathbf{u}\cdot\nabla\phi=\gamma(\Delta\phi-f(\phi)+\xi(t)), \tag{3.5c}$$

where ρ is the density of mixture, i.e.,

$$\rho=\frac{1+\phi}{2}\rho_1+\frac{1-\phi}{2}\rho_2$$

(ρ_1 and ρ_2 being the densities for two components of the fluid), $\mathbf{u}$ represents the velocity vector of the fluids, p is the pressure, λ corresponds to the surface tension and $\mathbf{g}$ is the gravitational acceleration. $\xi(t)$ is the Lagrange multiplier corresponding to the constant volume constraint on the physical domain $\Omega\subset\mathcal{R}^n$:

$$\frac{d}{dt}\int_\Omega\phi dx=0. \tag{3.6}$$

3.1 A modification to the Allen-Cahn equation

The role of the Lagrange multiplier $\xi(t)$ in (3.5c) is to change the constant value of the phase function ϕ. The constant values, suppose to be 1 or -1, represent two phase fields which are very important for the model. It is natural to adjust the value in the diffusive field of ϕ to conserve the volume constant. In (3.5c), the term $\xi(t)$ is changed into $\xi(t)(1-\phi^2)$. Such a scheme will keep the maximum principle for ϕ. By integration of the equation (3.5c), we get the explicit formula for the new $\xi(t)$:

$$\xi(t)=\int_\Omega f(\phi)dx/\int_\Omega(1-\phi^2)dx. \tag{3.7}$$

3.2 Finite element scheme

Together with the initial and boundary conditions,

$$\mathbf{u}(x,0)=\mathbf{u}_0(x),\quad \phi(x,0)=\phi_0(x),$$

$$\mathbf{u}(x,t)\Big|_{\partial\Omega}=0,\quad \frac{\partial\phi}{\partial\mathbf{n}}(x,t)\Big|_{\partial\Omega}=0,$$

the Navier-Stokes equation and phase field equation (3.5) determine the velocity $\mathbf{u}$, the pressure p and the interface ϕ. The discretization of the time derivative is given by the backward difference formula:

$$\frac{\partial \mathbf{u}}{\partial t}=\frac{3\mathbf{u}^{n+1}-4\mathbf{u}^{n}+\mathbf{u}^{n-1}}{2\Delta t}+\mathcal{O}((\Delta t)^2), \tag{3.8}$$

resulting in a semi-implicit second-order approximation. Introduce the notation

$$\mathbf{N}(\mathbf{u},\phi)=-\rho\mathbf{u}\cdot\nabla\mathbf{u}-\lambda\nabla\cdot(\nabla\phi\otimes\nabla\phi)+\rho\mathbf{g}(x). \tag{3.9}$$

Suppose $\mathbf{u}^n$, p^n and ϕ^n are known. Using the consistent splitting scheme [17], the variational formulation reads: $\forall\mathbf{v}\in H_0^1(\Omega)^2$ and $\forall q\in H^1(\Omega)$,

$$\left(\rho\frac{3\mathbf{u}^{n+1}-4\mathbf{u}^{n}+\mathbf{u}^{n-1}}{\Delta t},\,\mathbf{v}\right)+(\tau,\,\nabla\mathbf{v})=2(\mathbf{N}^{n}-\mathbf{N}^{n-1},\,\mathbf{v})-(\nabla p^{n},\,\mathbf{v}), \tag{3.10}$$

$$(\nabla p^{n+1},\,\nabla q)=-\mu(\nabla\times\mathbf{u}^{n+1},\,\mathbf{n}\times\nabla q)_{\partial\Omega}+(\mathbf{N}^{n+1},\,\nabla q). \tag{3.11}$$

It follows from the identity

$$\Delta\mathbf{u}=\nabla\nabla\cdot\mathbf{u}-\nabla\times\nabla\times\mathbf{u},$$

that this strategy yields a priori control on the divergence of $\mathbf{u}^{n+1}$, see, [17]. It is noted that

$$(\nabla\times\nabla\times\mathbf{u},\,\nabla q)=-(\nabla\times\mathbf{u},\mathbf{n}\times\nabla q)_{\partial\Omega}$$

is used to avoid computing $\nabla\times\nabla\times\mathbf{u}$.

We then update ϕ^{k+1} by the following procedure: $\forall\psi\in H^1(\Omega)$

$$\begin{aligned}&\left(\frac{3\phi^{n+1}-4\phi^{n}+\phi^{n-1}}{2\Delta t},\,\psi\right)+\gamma\left(\nabla\phi^{n+1},\,\nabla\psi\right)\\&=\left(\mathbf{u}^{n+1}\cdot\nabla(2\phi^{n}-\phi^{n-1}),\,\psi\right)-\gamma\left((2f(\phi^{n})-f(\phi^{n-1}))+\xi^{n+1}(1-(\phi^{n})^2),\,\psi\right).\end{aligned} \tag{3.12}$$

In [17], numerical experiments suggest that (3.10)-(3.11) is well-posed for the $\mathcal{P}_N/\mathcal{P}_N$ (but not necessarily inf-sup stable) approximations. (However, the theoretical justification of this statement is still open.) In our computations, piecewise linear polynomials are used for the spatial discretization of all variables $\mathbf{u}$, p and ϕ.

4 Numerical results

The idea that a fluid-fluid interface is diffuse, that is, has finite thickness, goes back to Poisson (1831) and Gibbs (1876) who recognized that the interface actually represented a rapid but smooth transition of physical quantities between the bulk fluid values. In this spirit, the so-called phase field function (the order parameter) is introduced to characterize the different bulk fluids. It assumes distinct constant values in each bulk phase

and undergoes rapid but smooth variation in the interfacial region. Based on energetic variational formulations, the diffuse interface models were developed. Recently, diffuse interface modeling of multi-phase mixtures has become fairly popular as a numerical device in many instances, see, e.g., [1, 4, 34].

The phase field models share common features with a number of methods developed from a more computational point of view, especially the level set method [24], which is a highly successful computational scheme applied to interfacial motion. With this method, the interface is represented as a level set of a smooth auxiliary function that is computationally analogous to the order parameter in phase field models. An advantage of the level set method is that the interface remains sharp in this formulation. However, after the advection of the interface, it is uncommon for the level set function to remain a signed distance function, which means that a normalization procedure needs to be done in order to limit numerical dissipation.

4.1 Modified Allen-Cahn phase equation: revisit

Our moving mesh algorithm is dimension independent. To demonstrate this, we begin by repeating Example 1 of [21]. The physical parameters are

$$\eta = 0.02, \quad \lambda = 0.1, \quad \nu = 0.1, \quad \gamma = 0.1,$$

the computational parameters are $\Omega = [0,2\pi]\times[0,2\pi]$, $dt = 0.005$, $\alpha = 10$, and a moving mesh containing 64×64 nodes is used. This test is used to exhibit the surface tension effects of the phase field model. It starts with a cubic bubble, and $\phi = 1$ inside the bubble and $\phi = -1$ outside the bubble. The cubic bubble quickly deforms into a circular bubble due to the surface tension and the volume of the bubble is preserved. It can be seen from Fig. 1 that our results are almost the same as those in [21]. Fig. 2 presents the maximum of the phase field function by using the multiplier of [32] and our modified multiplier (3.7). It is observed that the new formulation conserves the maximum norm of the phase function better than the old one.

4.2 Coalescence of two kissing bubbles

Consider two kissing bubbles of the same size. This problem has been considered in [21]. In this case the physical parameters are set to be

$$\eta = 0.02, \quad \lambda = 0.1, \quad \nu = 0.1, \quad \gamma = 0.1.$$

The computational parameters are $\Omega = 2\times 2\times 4$, $dt = 0.002$, $\alpha = 10$, and a moving mesh containing $32\times 32\times 64$ nodes is used. It is noted that η is the capillary width (mixing region) of the fluids, λ/η is the surface tension constant, ν is the viscosity and γ is the "elastic" relaxation time. Fig. 3 shows the shape evolution of two kissing bubbles and the meshes on the central slice at the indicated time instances. As time evolves, the two

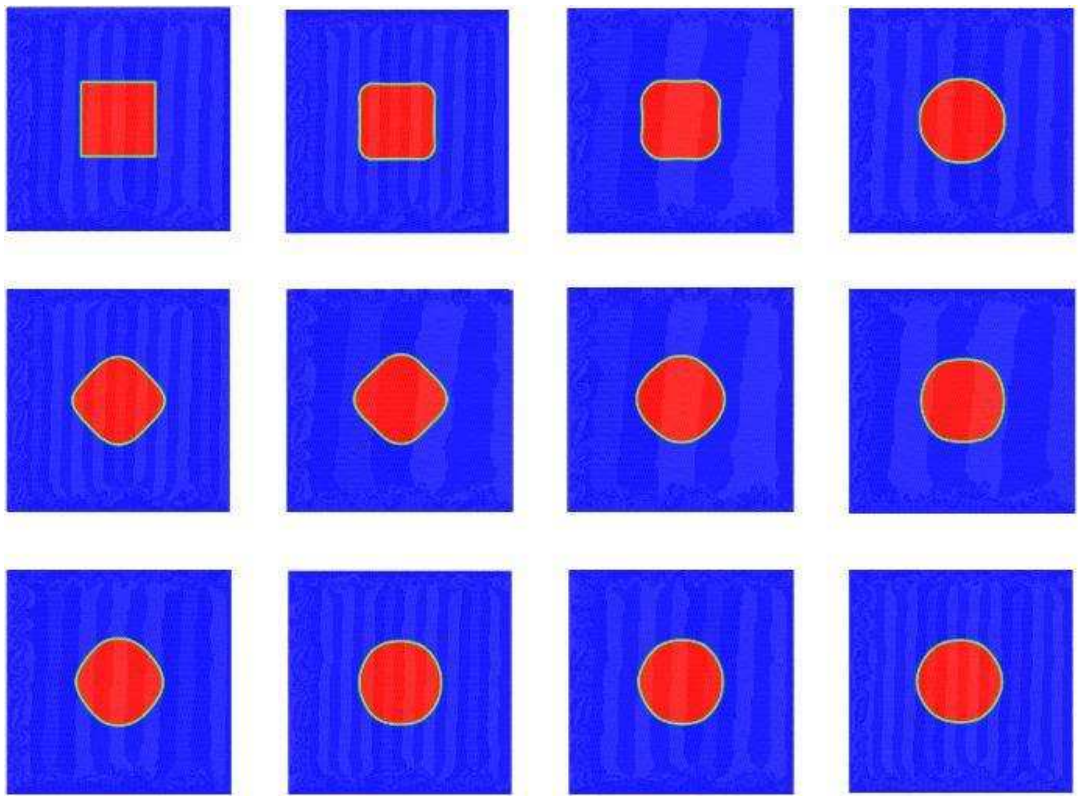


Figure 1: Re-visit of Example 1 of [21]: phase evolution of a cubic bubble at $t=0$, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 1.2, 1.4, 1.7, 2.5.

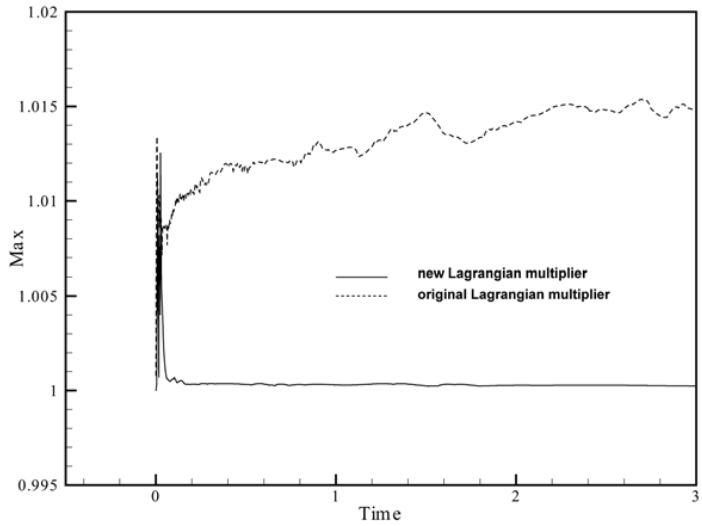


Figure 2: Re-visit of Example 1 of [21]: $\|\phi(\bullet,t)\|_\infty$ computed by using the original multiplier and the modified version (3.7).

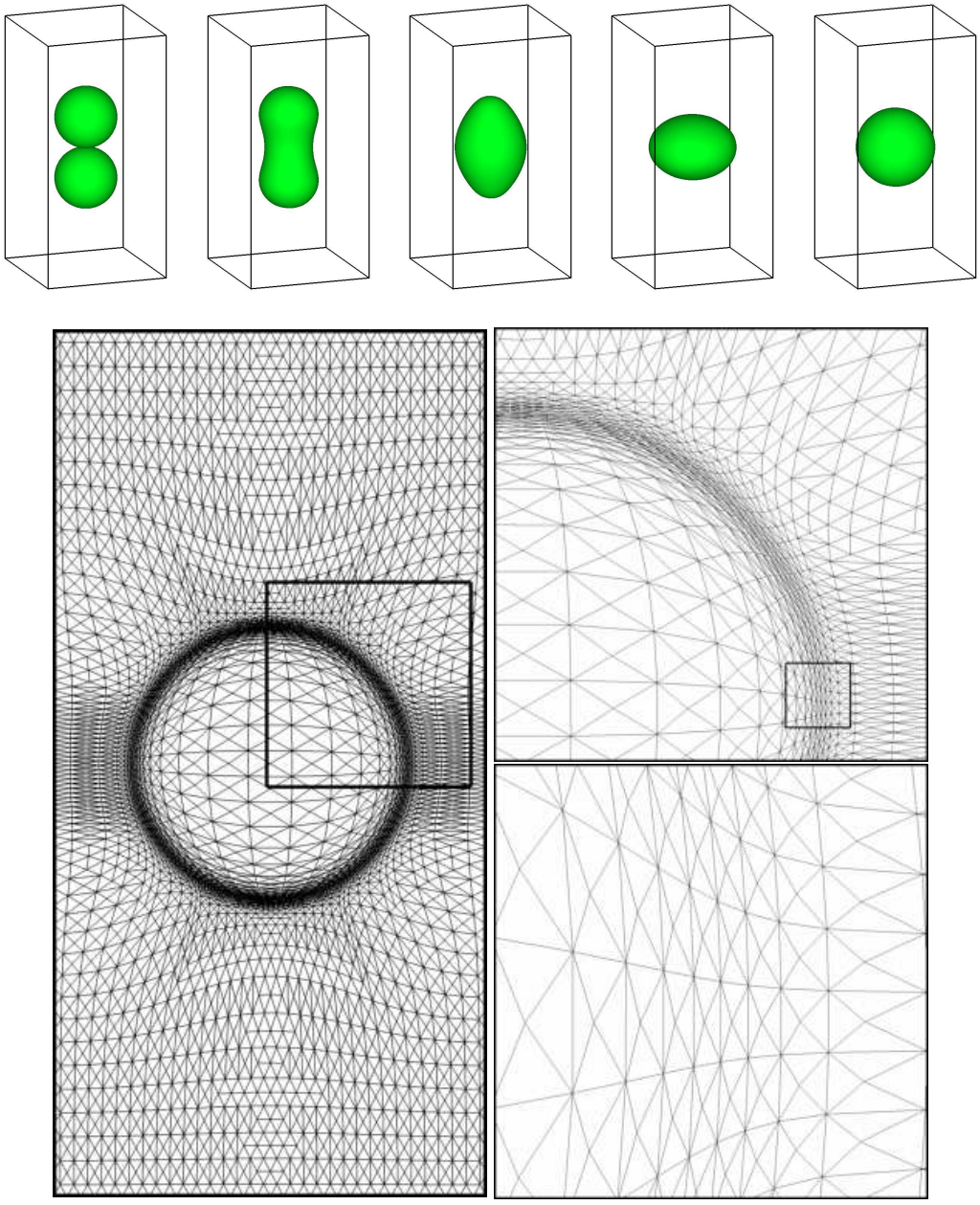
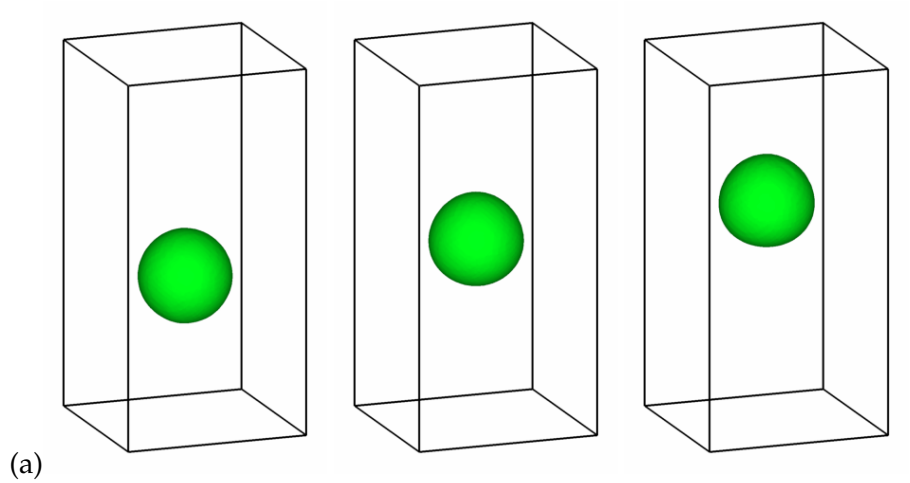
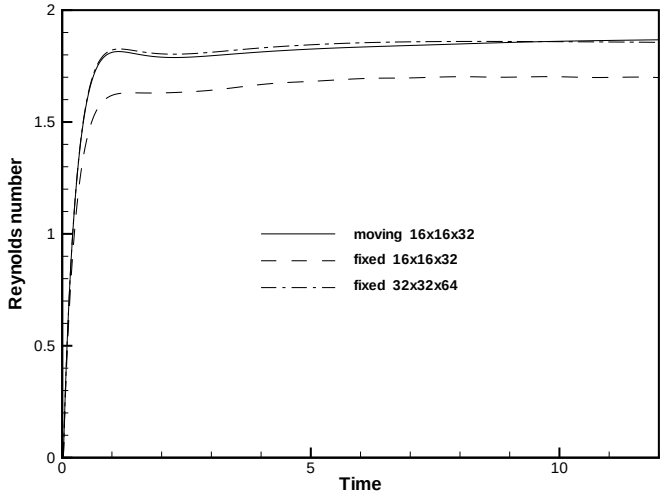


Figure 3: Coalescence of two kissing bubbles, $\eta=0.04$, $\lambda=0.1$, $\nu=0.1$, $\gamma=0.1$, $32\times32\times64$ grid in an $1\times1\times2$ domain. $t=0,0.1,0.3,0.6,0.8$. The bottom figures show the computational meshes ($t=0.8$). The box regions are magnified in the next figure.

bubbles coalesce into one big elliptic bubble. Then it oscillates between an ellipsoid and a sphere, and eventually stabilizes as one circular bubble. This is the combination of the surface tension effect and the elastic effect from the phase equation. Fig. 3 also shows



(a)



(b)

Figure 4: Rising bubble example: (a) the bubble evolution: $t=1,4,7$ (b) the rising Reynolds number, obtained with two fixed grids ($16\times16\times32$, $32\times32\times64$) and one moving grid ($16\times16\times32$).

computational mesh at $t=0.8$ which is locally magnified. On one hand, it can be seen that most of the mesh nodes are clustered in the region of the interface. On the other hand, it is clearly observed that the meshes obtained by using our method are locally uniform and globally smooth. As a result, a good approximation of the phase function near the interface is obtained.

4.3 Bubble rising

The rising bubble problem is a classical example for validating multi-fluid flow simulations, see, e.g., [10, 30]. Bubbles with lower density than the surrounding fluid tend to

rise, due to the buoyancy effects resulting from the pressure gradient caused by gravity. For high surface tension, $\lambda=\sigma\eta/\rho_0 gD^2$, bubbles remain nearly spherical and rise in a steady-state fashion with constant velocity and shape. In terms of the non-dimensional numbers, we set $\eta=0.04$, $\lambda=0.04$, $\nu=1.0/10^{3/4}$, $\gamma=0.01$, $\rho_o/\rho_b=20$, $\mu_0/\mu_b=20$ (same parameters as used in Fig. 1(b) of [30]). These non-dimensional parameters can, for example, be realized using air bubbles of diameter 1.9 mm in the standard engine oil (taking $\sigma=0.03N\,m^{-1}$, $\rho=880kg\,m^{-3}$, and $\mu=0.21N\,s\,m^{-2}$), with the exception that the density and viscosity ratios would be lower. The $1\times1\times2$ domain is covered by a moving mesh containing $16\times16\times32$ nodes. The time step used is $dt=0.002$.

For the three-dimensional bubbles, the bubble volume is approximated by

$$V_b=\int_{v_b}d\mathbf{x}=\int_\Omega\frac{1}{2}(1+\phi)d\mathbf{x}. \tag{4.1}$$

The centroid and the vertical centroid are found respectively by

$$Z_c=\frac{1}{V_b}\int_{v_b}zd\mathbf{x}=\int_\Omega\frac{1}{2}(1+\phi)zd\mathbf{x},\qquad W_b=\frac{1}{V_b}\int_{v_b}wd\mathbf{x}=\int_\Omega\frac{1}{2}(1+\phi)wd\mathbf{x}. \tag{4.2}$$

In Fig. 4, we plot the evolution of the rising bubble and the rising Reynolds numbers with three different grid resolutions. The Reynolds number is defined by

$$Re=\rho_f DW_b/\mu_f=W_b/\nu. \tag{4.3}$$

The accuracy of the moving method is checked by comparing its result with fixed grid results. On a coarse $16\times16\times32$ grid, it is observed that the initial unsteady motion is well resolved, both on fixed and adaptive grid, but the agreement is lost after $t\geq0.5$. Results for a fixed $32\times32\times64$ grid simulation are also included, which indicate that the moving mesh result on the coarse mesh has essentially converged.

4.4 Non-axisymmetric merging of two bubbles

In Fig. 5, we display the interaction of two viscous gas bubbles in a liquid. The density ratio is 20:1 and the viscosity ratio is 26:1. The dimensionless parameters we use for this problem are

$$\eta=0.04,\quad \lambda=\eta/We=0.02\eta,\quad \nu=1/50^{3/4},\quad \gamma=0.01,\quad Fr=1.$$

The domain has a dimensionless size $4\times4\times8$, and the basic tetrahedral mesh contains $32\times32\times64$ nodes. For this problem, we start off with two spherical bubbles whose centers are offset in the "x" direction by one bubble radii R and offset in the "z" direction by $2.3R$. This setting is the same as the one used by [25,30]. In Fig. 6, the left plot shows the actual elements neighboring the plane $y=2$, and the right one shows the actual elements neighboring the surface of the bubbles.

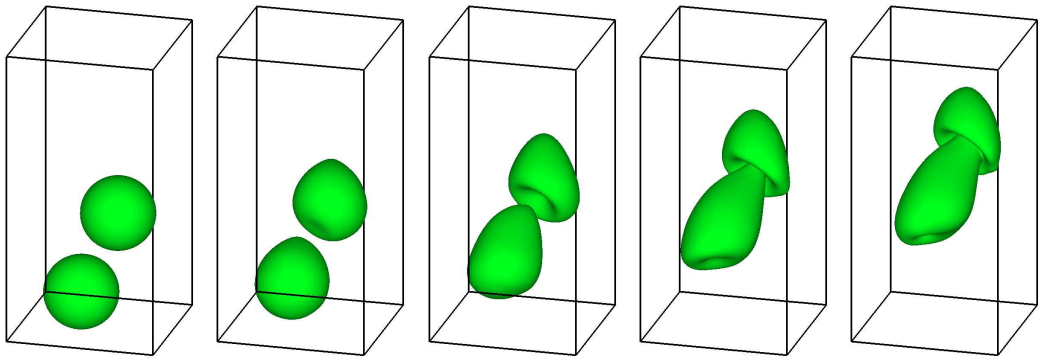


Figure 5: Non-axisymmetric merging of two viscous gas bubbles ($\eta=0.04$, $\lambda=\eta/We=0.02\eta$, $\nu=1/50^{3/4}$, $\gamma=0.01$ and $Fr=1$). The moving tetrahedral mesh contains $32\times32\times64$ nodes. $t=0.0,1.0,2.0,3.0,4.0$.

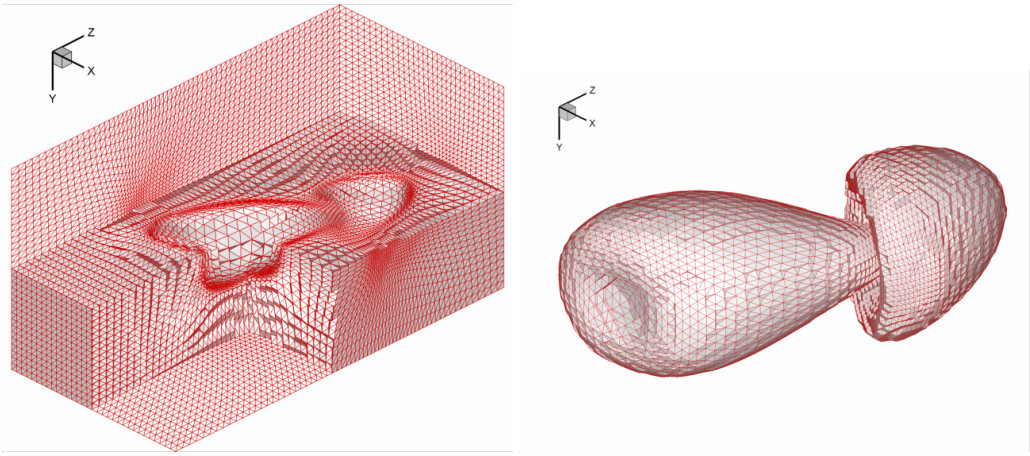


Figure 6: Non-axisymmetric merging of two viscous gas bubbles: the moving tetrahedral mesh contains $32\times32\times64$ nodes. $t=3.0$.

As the bubbles rise they deform considerably. The bottom of the top bubble folds upward and deforms the lower bubble into a "pear" shape, pointing toward the top one. The top bubble continues to deform and draws the lower bubble. This observation is in qualitative agreement with [30]. Although we have calculated the process for a longer time, the upward suction by the top bubble is not so strong as [30]. Our result seems quite similar to that given by [25]. The interfaces in [30] (front tracking method) and [25] (level set method) remain sharp. In contrast, formulating a phase-field model requires an asymptotic expansion analysis be performed with a small parameter proportional to the interface width η. The phase-field method has an approximate representation of the front location, resulting possible difference with the front tracking or level set modelings.

In Fig. 7, we compare the results obtained on moving mesh grids with $32\times32\times64$

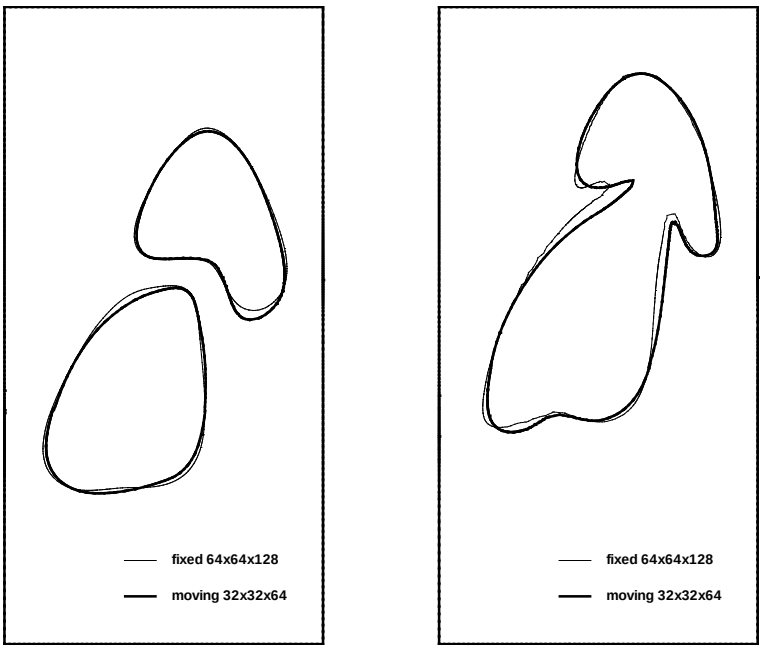


Figure 7: Non-axisymmetric merging of two viscous gas bubbles: the interface in a slice ($y=1$, $\mathbf{n}=(0,1,0)$) obtained by the uniform tetrahedral mesh with $64\times64\times128$ cells (thin) and the moving tetrahedral mesh with $32\times32\times64$ cells (thick). $t=2.0$ (left) and 3.0 (right).

cells and on fixed grids with $64\times64\times128$ cells. We observe a good agreement between the two results.

5 Concluding remarks

In this work, we present an adaptive moving mesh algorithm for meshes of unstructured tetrahedra in three dimensions. The algorithm automatically adjusts the size of the elements with time and position in the physical domain to resolve the relevant scales in multiscale physical systems while minimizing computational costs. To demonstrate the performance of the proposed 3D moving mesh strategy, the algorithm is implemented in finite element simulations of deformable droplet and fluid-fluid interface interactions in multiphase flows. There have been many proposed models and numerical computations for simulating interfacial dynamics in filament breakdown and drop formulation, including the level set approach and phase-field approach. However, most of the simulations have been restricted to two space dimension computations due to large computational cost involved. In this work, we propose a general framework on designing an adaptive moving grid method useful for this kind of simulations.

Although the PDE time-evolution algorithm used in this work is based on the finite element approach, it should be pointed out that there have been extensive studies of

moving mesh algorithms based on finite difference approaches, see, e.g., [3,9,28]. Moving mesh methods based on finite difference (or finite volume) methods enjoy the simplicity in coding and are more appropriate for problems where finite difference methods have shown advantages (such as nonlinear hyperbolic problems). Apart from the flexibility of solution domains, another reason for using finite elements but not finite differences in this work is that with finite element methods we can avoid using pointwise interpolation which was used in many moving mesh finite difference methods. Instead a more natural continuous type solution interpolation procedure can be used.

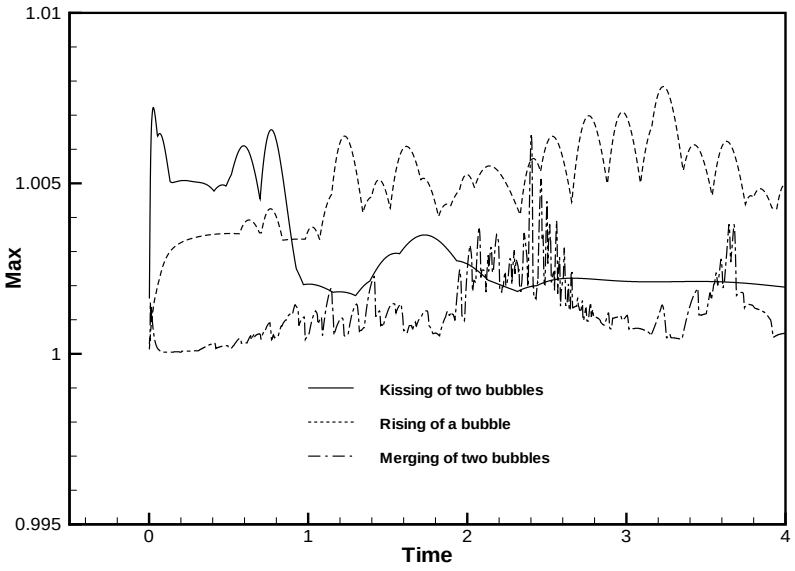


Figure 8: The maximum of the phase function is well preserved.

We close by discussing two issues related to the models and the numerical methods used in this work. The first issue is the preservation of the magnitude of the phase field function ϕ, which should satisfy $\|\phi\|_\infty = 1$. In Fig. 8, we plot $\|\phi\|_\infty = 1$ for the last three test problems, with the modified formula (3.7). The results are found very satisfactory.

The second issue is the merit of the use of moving mesh methods. We compared the efficiency of the moving mesh methods for the nonaxisymmetric merging of two bubbles in Section 4.4. The comparisons are based on two aspects: the CPU time and the storage used for the same resolution. To reach similar accuracy, the fixed mesh simulation requires at least $64\times64\times128$ cells to match that for a moving mesh with $32\times32\times64$ cells. In other words, a total of 8 times storage saving is gained. We point out that the storage usage is less important in 2D computations, but it becomes very important in 3D simulations. To reach $t=4$, the simulation on the moving grid with $32\times32\times64$ cells takes about 20h 30min in a Xeon 3.0GHz machine, while the simulation on the $64\times64\times128$ fixed mesh takes about 106h 20min.

Acknowledgments

The research of Di and Li was supported in part by the Joint Applied Mathematics Research Institute of Peking University and Hong Kong Baptist University. Li was also partially supported by the National Basic Research Program of China under the grant 2005CB321701. The research of Tang was supported by CERG Grants of Hong Kong Research Grant Council, FRG grants of Hong Kong Baptist University, and NSAF Grant #10476032 of National Science Foundation of China. He was supported in part by the Chinese Academy of Sciences while visiting its Institute of Computational Mathematics.

References

[1] D. M. Anderson, G. B. McFadden and A. A. Wheeler, Diffuse-interface methods in fluid mechanics, Annu. Rev. Fluid Mech., 30 (1998), 139-165.
[2] E. Aulisa, S. Manservisi and R. Scardovelli, A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows, J. Comput. Phys., 188 (2003), 611-639.
[3] B. N. Azarenok and T. Tang, Second-order godunov-type scheme for reactive flow calculations on moving meshes, J. Comput. Phys., 106 (2005), 48-80.
[4] G. Beckett, J. A. Mackenzie and M. L. Robertson, An r-adaptive finite element method for the solution of the two-dimensional phase-field equations, Commun. Comput. Phys., 1 (2006), 805-826.
[5] J. U. Brackbill, An adaptive grid with directional control, J. Comput. Phys., 108 (1993), 38-50.
[6] J. U. Brackbill and J. S. Saltzman, Adaptive zoning for singular problems in two dimensions, J. Comput. Phys., 46 (1982), 342-368.
[7] A. Brandt, S. F. McCormick and J. W. Ruge, Algebraic multigrid (AMG) for sparse matrix equations, in: D. J. Evans (Ed.), Sparsity and its Applications, Cambridge University Press, Cambridge, 1984.
[8] W. M. Cao, W. Z. Huang and R. D. Russell, A study of monitor functions for two dimensional adaptive mesh generation, SIAM J. Sci. Comput., 20 (1999), 1978-1994.
[9] H. D. Ceniceros and T. Y. Hou, An efficient dynamically adaptive mesh for potentially singular solutions, J. Comput. Phys., 172 (2001), 609-639.
[10] F. S. de Sousa, N. Mangiavacchi, L. G. Nanato, A. Castelo, M. F. Tome, V. G. Ferreira, J. A. Cuminato and S. McKee, A front-tracking/front-capturing method for the simulation of 3d multi-fluid flows with free surfaces, J. Comput. Phys., 198 (2004), 469-499.
[11] Y. Di, R. Li, T. Tang and P. W. Zhang, Moving mesh methods for singular problems on a sphere using perturbed harmonic mappings, SIAM J. Sci. Comput., 28 (2006), 1490-1508.
[12] Y. Di, R. Li, T. Tang, and P. W. Zhang, Moving mesh finite element methods for the incompressible Navier-Stokes equations, SIAM J. Sci. Comput., 26 (2005), 1036-1056.
[13] Y. Di, R. Li, T. Tang and P. W. Zhang, Level set calculations for incompressible two-phase flows on a dynamically adaptive grid, J. Sci. Comput., 31(1-2) (2006), 75-98.
[14] Y. Di and P. W. Zhang, Moving mesh kinetic simulation for sheared rodlike polymers with high potential intensities, Commun. Comput. Phys., 1 (2006), 859-873.
[15] A. S. Dvinsky, Adaptive grid generation from harmonic maps on Riemannian manifolds, J. Comput. Phys., 95 (1991), 450-476.

[16] D. Enright, R. Fedkiw, J. Ferziger and I. Mitchell, A hybrid particle level set methods for improved interface capturing, J. Comput. Phys., 183 (2002), 83-116.
[17] J. L. Guermond, P. Minev and J. Shen, An overview of projection methods for incompressible flows, Comput. Method. Appl. Mech. Engrg., 1 (2006), 1-41.
[18] M. Kang, R. Fedkiw and X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, J. Sci. Comput., 15 (2000), 323-360.
[19] R. Li, T. Tang and P.-W. Zhang, Moving mesh methods in multiple dimensions based on harmonic maps, J. Comput. Phys., 170 (2001), 562-588.
[20] R. Li, T. Tang and P.-W. Zhang, A moving mesh finite element algorithm for singular problems in two and three space dimensions, J. Comput. Phys., 177 (2002), 365-393.
[21] C. Liu and J. Shen, A phase field model for the mixture of two incompressible fluids and its approximation by a Fourie-spectral method, Physica D, 179 (2003), 211-228.
[22] M. S. Longuet-Higgins and E. D. Cocklet, Deformation of steep surface waves on water I: A numerical method of computation, Proc. R. Soc. Lond. A, 350 (1975), 1-25.
[23] F. Losasso, R. Fedkiw and S. Osher, Spatially adaptive techniques for level set methods and incompressible flow, Comput. Fluids, 35 (2006), 995-1010.
[24] S. Osher and J. A. Sethian, Fronts propagation with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations, J. Comput. Phys., 79 (1988), 12-49.
[25] M. Sussman and E. G. Puckett, A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows, J. Comput. Phys., 162 (2000), 301-337.
[26] M. Sussman, P. Smeraka and S. J. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys., 114 (1994), 146-159.
[27] H. Z. Tang, A moving mesh method for the euler flow calculations using a directional monitor function, Commun. Comput. Phys., 1 (2006), 656-676.
[28] H. Z. Tang and T. Tang, Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws, SIAM J. Numer. Anal., 41 (2003), 487-515.
[29] A.-K. Tornberg and B. Engquist, A finite element based level set method for multiphase flow applications, Comput. Visual. Sci., 3 (2000), 93-101.
[30] S. Unverdi and G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, J. Comput. Phys., 100 (1992), 25-37.
[31] A. Winslow, Numerical solution of the quasi-linear Poisson equation, J. Comput. Phys., 1 (1967), 149-172.
[32] X. Yang, J. J. Feng, C. Liu and J. Shen, Numerical simulations of jet pinching-off and drop formation using an energetic variational phase-field method, J. Comput. Phys., 218 (2006), 417-428.
[33] X. Yang, A. J. Lowengrub, X. Zheng and V. Cristini, An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids, J. Comput. Phys., 217 (2006), 364-394.
[34] P. Yue, J. J. Feng, C. Liu and J. Shen, A diffuse-interface method for simulating two-phase flows of complex fluids, J. Fluid Mech., 515 (2004), 293-317.
[35] Z. R. Zhang, Moving mesh method with conservative interpolation based on l^2-projection, Commun. Comput. Phys., 1 (2006), 930-944.