Mecánica Computacional Vol. XXIV
A. Larreteguy (Editor)
Buenos Aires, Argentina, Noviembre 2005

AN ARBITRARY LAGRANGIAN-EULERIAN LEVEL-SET METHOD FOR INCOMPRESSIBLE TWO-DIMENSIONAL TWO-FLUID FLOWS

Fabricio S. Sousa*and Norberto Mangiavacchi†

*Intituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação e Estatística Universidade de São Paulo, USP
Av. Trabalhador São Carlense, 400, Cx.P. 668, 13251-900, São Carlos, SP, Brasil.
email: fsimeoni@lcad.icmc.usp.br, web page: http://www.lcad.icmc.usp.br

†Faculdade de Engenharia
Departamento de Engenharia Mecânica Universidade Estadual do Rio de Janeiro, UERJ
Rua São Francisco Xavier, 524, 20550-900, Rio de Janeiro, RJ, Brasil.
email: norberto@uerj.br, web page: http://www.uerj.br

Key Words: Two-fluid flows, Incompressible flows, Arbitrary Lagrangian-Eulerian, Level-set, Moving mesh.

Abstract. *An Arbitrary Lagrangian-Eulerian (ALE) level-set method to solve incompressible two-dimensional two-fluid flows is presented. The Navier-Stokes equations are discretized by a Galerkin Finite Element method. A projection method based on approximated LU decomposition is employed to decouple the system of non-linear equations. The interface between fluids is represented by a discrete Heaviside function plus additional marker points and edges of the computational mesh. Our method employs a technique which moves the nodes of the Finite Element mesh with arbitrary velocity. The quality of the mesh is controlled by a remeshing procedure, avoiding bad triangles by flipping edges, inserting or removing vertices from the triangulation. The relative velocity in the ALE approach is designed to allow for a continuous improvement of the mesh, thus reducing the amount of remeshing required to control the quality of the mesh. Results of numerical simulations are presented, illustrating the improvements in computational cost, mass conservation, and accuracy of this new methodology.*

1 INTRODUCTION

Simulations of multi-fluid flows are known to be difficult to perform due to discontinuities at the fronts separating the different fluids. A number of methods have been developed to approximate the fronts, and they can be classified in two main groups: Front-tracking and Front-capturing methods.[1] In Front-tracking methods, the fronts are represented by computational elements like marker particles, that move through the domain with the fluid velocity field. A number of papers dealing with Front-tracking methods can be found in the literature.[2–4] The Front-tracking methodology is more accurate than Front-capturing, introducing very small mass variation of the fluids involved in the simulation. However, its implementation is more difficult, in particular when the flows undergo topological changes, like either coalescence or splitting of the interfaces. On the other hand, Front-capturing methods represent the interfaces by a region of high gradient variation, where the fronts are reconstructed at each time-step. Among these, the level-set method, introduced by Osher and Sethian,[5] has acquired popularity because of its algorithmic simplicity. In this method, the fronts are represented by the zero level set of a function, that is advected by solving $\phi_t + \mathbf{u} \cdot \nabla\phi = 0$ where $\mathbf{u}$ is the velocity field. Most numerical procedures designed to solve this equation will introduce artificial diffusion leading to pronounced mass conservation errors.

The above described methods can be applied to both structured and unstructured grids. Methods employing unstructured grids can produce more accurate and efficient methods by selectively refining regions of the domain where the interface and other important small scale features of the flow occur. For instance, Chen, Minev and Nandakumar[6] present a finite element method for incompressible multiphase flows with capillary interfaces on a fixed Eulerian grid, in which the fluid phases are identified and advected using a level set function, and the grid is temporarily adapted around the interfaces. The resulting technique can be considered as a compromise between the arbitrary-Lagrangean-Eulerian (ALE) approach and the fixed grid, Eulerian approach. Perot and Nallapati[7] developed a front tracking method for free-surface flows that employs an unstructured mesh that dynamically adjusts to the free surface. The points in the interior of the domain do not move in a Lagrangian fashion to avoid strong distortion of the mesh. Instead, each edge of the mesh is treated as a linear spring, requiring the solution of an equilibrium equation at each step. The flow field is then updated using an Arbitrary Lagrangian Eulerian (ALE) formulation. The proposed method, however, does not deal with splitting or reconnections of the free surface.

In order to minimize mass conservation errors and improve accuracy of the level-set technique for incompressible two-phase flows, a new approach was proposed by Sousa and Mangiavacchi[8] which can be classified as a Lagrangian level-set method. It uses the level-set of a pseudo-concentration function to represent the fronts, but its advection is performed by moving the nodes of the mesh, where the values of this function are stored. The method thus has features of both front tracking and front capturing methods, and can take advantage of them. In this work, the Lagrangian method employed by Sousa and Mangiavacchi is extended to allow for the relative movement between fluid particles and mesh points, in an ALE apporach. The relative velocity is designed to improve the quality of the mesh, thus reducing the amount of insertions and deletions of nodes required to mantain the quality of the mesh. In the following sections, the formulation, discretization and underlying numerical method employed in the proposed unstructured mesh ALE approach for the simulation of multiphase and free surface flows will be shortly described. Some implementation issues regarding the interface discretization will be discussed.

2 FORMULATION

The conservation equations modeling incompressible multi-fluid flows are the equation of motion in a moving computational mesh

$$\frac{\partial(\rho\mathbf{u})}{\partial t}+((\mathbf{u}-\hat{\mathbf{u}})\cdot\nabla)\rho\mathbf{u}=-\nabla p+\frac{1}{Re}\nabla\cdot\left[\mu\left(\nabla\mathbf{u}+\nabla\mathbf{u}^T\right)\right]+\frac{1}{Fr^2}\,\rho\mathbf{g}+\frac{1}{We}\mathbf{f}\,, \quad (1)$$

and the continuity equation $\nabla\cdot\mathbf{u}=0$, where $\mathbf{u}$ is the velocity field, p is the pressure field, μ and ρ are the discontinuous viscosity and density, $\mathbf{g}$ represents the gravitational acceleration field and $\mathbf{f}$ is a source term representing the surface tension. In this equation, Re, Fr and We are the non-dimensional Reynolds, Froude and Weber numbers. Using the CSF model,[9] the source term can be written as $\mathbf{f}=\sigma\kappa\nabla H$, where σ is the surface tension coefficient, κ is the curvature and H is a Heaviside function which is 1 inside one fluid and 0 outside. Here, $\hat{\mathbf{u}}$ is the mesh velocity which is computed as a combination between fluid velocity and an elastic velocity, more specifically

$$\hat{\mathbf{u}}=\beta_1\mathbf{u}+\beta_2\mathbf{u}_e \quad (2)$$

where $\mathbf{u}_e$ is an elastic velocity computed based on a Laplacian filter applied to the node positions, in order to improve the quality of the elements. In eq. (2), parameters β_1 and β_2 are chosen between 0 and 1 to control the final velocity. Notice that $\beta_1=1$ for the interface nodes, where they move in a Lagrangian fashion.

3 DISCRETIZATION

The domain is discretized by an unstructured triangular mesh which is initially a Delaunay triangulation. The element used in this approximation is the mini-element ($P1^+-P1$), with 4 velocity nodes (one in each vertex plus one in the centroid of the triangle) and 3 pressure nodes (one in each vertex). The shape functions interpolating the discrete approximations are assumed to be linear plus a bubble function in the centroid for the velocity, and linear for pressure, both continous. Considering $\mathbb{V}=H^1(\Omega)^m=\{\mathbf{v}=(v_1,\ldots,v_m):v_i\in H^1(\Omega),\forall\,i=1,\ldots,m\}$, where $H^1(\Omega)$ is a Sobolev space, and the sub-spaces $\mathbb{V}_{\mathbf{u}_\Gamma}=\{\mathbf{v}\in\mathbb{V}:\mathbf{v}=\mathbf{u}_\Gamma\text{ em }\Gamma_1\}$, $\mathbb{P}_{p_\Gamma}=\{q\in L^2(\Omega):q=p_\Gamma\text{ em }\Gamma_2\}$, the weak formulation of the problem can be written as: find

$\mathbf{u}(\mathbf{x},t) \in \mathbb{V}_{\mathbf{u}_\Gamma}$ and $p(\mathbf{x},t) \in \mathbb{P}_{p_\Gamma}$ such that

$$\begin{aligned} m(\frac{\partial(\rho\mathbf{u})}{\partial t},\mathbf{w}) + a(\mathbf{u}-\hat{\mathbf{u}},\rho\mathbf{u},\mathbf{w}) - g(p,\mathbf{w}) + \frac{1}{Re}\, k(\mu,\mathbf{u},\mathbf{w}) & \\ -\frac{1}{Fr^2}\, m(\rho\mathbf{g},\mathbf{w}) - \frac{1}{We}\, m(\mathbf{f},\mathbf{w}) &= 0 \qquad (3) \\ d(q,\mathbf{u}) &= 0, \qquad (4) \end{aligned}$$

for all $\mathbf{w} \in \mathbb{V}_\mathbf{0}$ and $q \in \mathbb{P}_0$, where the functionals in (3)-(4) are given by

$$\begin{aligned} m(\mathbf{v},\mathbf{w}) &= \int_\Omega \mathbf{v}\cdot\mathbf{w}\, d\Omega \qquad (5) \\ a(\mathbf{u},\mathbf{v},\mathbf{w}) &= \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{v}\cdot\mathbf{w}\, d\Omega \qquad (6) \\ k(\mu,\mathbf{v},\mathbf{w}) &= \int_\Omega \mu\left[(\nabla\mathbf{v}+\nabla\mathbf{v}^T) : \nabla\mathbf{w}^T\right]\, d\Omega \qquad (7) \\ g(p,\mathbf{w}) &= \int_\Omega \nabla p\cdot\mathbf{w}\, d\Omega \qquad (8) \\ d(p,\mathbf{w}) &= \int_\Omega (\nabla\cdot\mathbf{w})p\, d\Omega \qquad (9) \end{aligned}$$

The discretization of (3)-(4) using the shape functions for the mini-element and Galerkin weighting, results in the following ODE system

$$\begin{gathered} \mathbf{M}_\rho\dot{\boldsymbol{u}} + \mathbf{A}\boldsymbol{u} + \tfrac{1}{Re}\mathbf{K}\boldsymbol{u} - \mathbf{G}\boldsymbol{p} - \tfrac{1}{Fr^2}\mathbf{M}_\rho\boldsymbol{g} - \tfrac{1}{We}\mathbf{M}\boldsymbol{f} = \mathbf{0} \\ \mathbf{D}\boldsymbol{u} = \mathbf{0}\ , \qquad (10) \end{gathered}$$

In 10, the quantities ***u***, ***v***, ***p***, ***g***, and ***f*** are vectors of dimension s, where s is the total number of nodes in the discretized domain. The ODE's are discretized in time using a semi-implicit time integration technique, where the non-linear terms are kept explicit. Thus, the matrices to be solved are symmetric positive definite and fast linear system solvers can be applied, e.g. the conjugate gradient method. Using a semi-implicit time integration, eq. (10) becomes

$$\begin{gathered} \mathbf{M}_\rho\left(\tfrac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^n}{\Delta t}\right) + \mathbf{A}\boldsymbol{u}^n + \tfrac{1}{Re}\mathbf{K}\boldsymbol{u}^{n+1} - \mathbf{G}\boldsymbol{p}^{n+1} - \tfrac{1}{Fr^2}\mathbf{M}_\rho\boldsymbol{g} - \tfrac{1}{We}\mathbf{M}\boldsymbol{f} = \mathbf{0} \\ \mathbf{D}\boldsymbol{u}^{n+1} = \mathbf{0}\ . \qquad (11) \end{gathered}$$

4 NUMERICAL METHOD

The numerical procedure employed to solve the Navier-Stokes equations is a projection method based in approximated block *LU* decomposition.[10–12] The system of discrete equations (11) can be rewriten as

$$\begin{bmatrix} \mathbf{B} & -\Delta t\mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{n+1} \\ \boldsymbol{p}^{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}^n \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{bc}_1 \\ \boldsymbol{bc}_2 \end{bmatrix} \qquad (12)$$

where $\boldsymbol{u}^{n+1} = [u_1^{n+1}, \ldots, u_{Nu}^{n+1}, v_1^{n+1}, \ldots, v_{Nv}^{n+1}]^T$, $p^{n+1} = [p_1^{n+1}, \ldots, p_{Np}^{n+1}]^T$, with Nu, Nv and Np being the number of free-nodes for the velocity (x and y direction) and pressure. The matrix **B** is given by

$$\mathbf{B} = \mathbf{M}_\rho + \frac{\Delta t}{Re}\mathbf{K} \tag{13}$$

and the right hand side are the known values at time n,

$$\boldsymbol{r}^n = -\Delta t \left(\mathbf{A}\boldsymbol{u}^n - \frac{1}{Fr^2}\mathbf{M}_\rho \boldsymbol{g} - \frac{1}{We}\mathbf{M}\boldsymbol{f} \right) + \mathbf{M}_\rho \boldsymbol{u}^n \ , \tag{14}$$

plus the contribution of the known values from the boundary conditions. Applying a canonical block *LU* decomposition[12] to the matrix of the system (12) results

$$\begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{D} & \Delta t \mathbf{D}\mathbf{B}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\Delta t \mathbf{B}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{n+1} \\ \boldsymbol{p}^{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}^n \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{bc}_1 \\ \boldsymbol{bc}_2 \end{bmatrix} \tag{15}$$

According to Chang et al.,[11] the method originating from eq. (15) is called Uzawa method. However this method is computationally expensive due to the inversion of matrix **B**. To avoid this operation, we substitute $\mathbf{B}^{-1}$ by an approximation $\mathbf{M}_{L,\rho}^{-1}$, which is a diagonal matrix, computed from the lumping of the mass matrix apearing in eq. (13), that can be easily inverted. This results in the following procedure

1. Solve $\tilde{\boldsymbol{u}}$ from $\mathbf{B}\tilde{\boldsymbol{u}} = \boldsymbol{r}^n + \boldsymbol{bc}_1$;
2. Solve $\boldsymbol{p}^{n+1}$ from $\Delta t \mathbf{D}\mathbf{M}_{L,\rho}^{-1}\mathbf{G}\boldsymbol{p}^{n+1} = -\mathbf{D}\tilde{\boldsymbol{u}} + \boldsymbol{bc}_2$;
3. Compute the final velocity from $\boldsymbol{u}^{n+1} = \tilde{\boldsymbol{u}} + \Delta t \mathbf{M}_{L,\rho}^{-1}\mathbf{G}\boldsymbol{p}^{n+1}$.

As we used the same approximation for $\mathbf{B}^{-1}$ in steps 2 and 3, the computed final velocity is divergence-free at discrete level.

5 INTERFACE REPRESENTATION

The interface between fluids is represented by vertices and edges belonging to the triangulation (see fig. 1), and is moved by the fluid velocity. Additionally, a discrete Heaviside function is also used to represent the interface, and its gradient is computed to distribute the interface tension force to the free-nodes where velocity is evaluated, as it was done in a work due to Sousa et al.[13] for structured grids. Since in this work the level-set function is transported in a Laplacian manner, with no artificial diffusion, the function does not need to be smooth as in the standard Eulerian level-set methods.

Hence, the discrete Heaviside function is defined as

$$H_{\lambda,j} = \begin{cases} 1, & \text{if vertex } j \text{ belongs to phase } \lambda \\ \frac{1}{2}, & \text{if vertex } j \text{ belongs to the interface of phase } \lambda \\ 0, & \text{if vertex } j \text{ does not belong to phase } \lambda \end{cases} \tag{16}$$

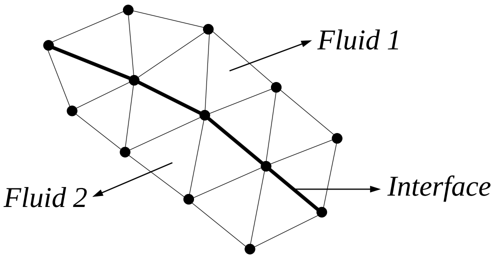

Figure 1: Representation of the interface using vertices and edges from the triangulation.

such that the interface is identified as the level-set $\frac{1}{2}$ of $H_{\lambda,j}$. Thus, the gradient of this function is used to distribute the interfacial force. In practice, it is not necessary to compute ∇H in the whole domain, but only in the neighborhood of the interface.

Discretizing **f** in a variational fashion, and applying the Galerking weighting, results

$$\mathbf{M}\boldsymbol{f}_D = \mathbf{\Sigma}\mathbf{G}\boldsymbol{h}_\lambda \ , \tag{17}$$

where $\mathbf{\Sigma}$ is a diagonal matrix with elements given by $\sigma\kappa_1, \ldots, \sigma\kappa_{NV}$, that are approximations for the intensity of the capilary pressure in the velocity nodes. Additionally, $\boldsymbol{h}_\lambda = [H_{\lambda,1}, \ldots, H_{\lambda,NP}]^T$ is the discrete *Heaviside* function computed in the pressure nodes. Notice that formally, κ is not defined away from the interface. We define κ in the velocity nodes adjacent to the interface as the weighted average from the known values at the interface nodes in the star of the velocity nodes. Finally, eq. (17) becomes

$$\boldsymbol{f}_D = \mathbf{M}^{-1}(\mathbf{\Sigma}\mathbf{G}\boldsymbol{h}_\lambda) \tag{18}$$

that can replace ***f*** in eq. (14).

6 MESH CONTROL

As the mesh is moved, elements can become distorted, resulting in bad elements to the finite element approximation. To avoid bad elements to appear in the mesh, a mesh control procedure is employed. Insertions and deletions of nodes are periodically made in the mesh, in order to remove elements considered bad from the triangulation.

A bad triangle can be classified in two types: a cap triangle, possesses a large circumscribing circle radius comparing to its edges. It can be removed by flipping the largest edge, inserting a new point in the middle of the largest edge or deleting the vertex which possesses the largest angle; a thin triangle, possesses a very small edge comparing to the other edges and to the circumscribing circle radius. It can be removed from the triangulation by deleting the shortest edge or inserting a point in the largest edge.

To garantee the quality of the resulting mesh, the vertices and edges are inserted into and removed from the mesh such that the resulting triangulation is Delaunay in the vicinity of the changed region. Although the entire mesh is not Delaunay, this ensures that the local triangulation is optimal.

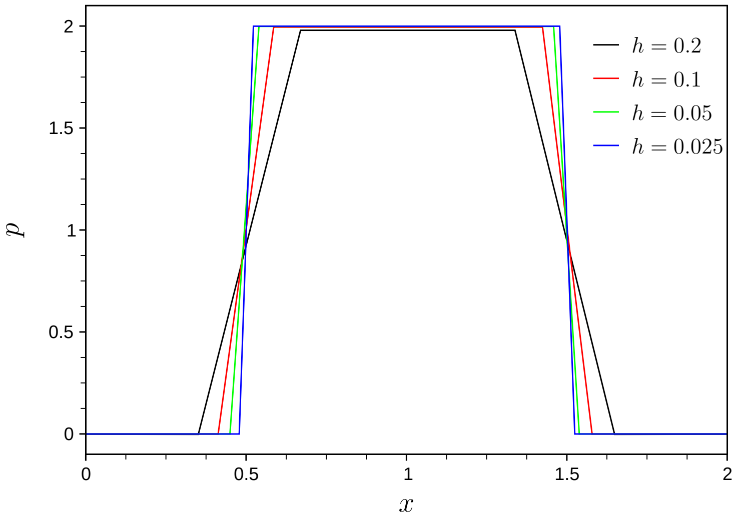

Figure 2: Pressure profiles in the static bubble simulation: flat profiles were obtained using the proposed force distribution.

Additionally, the elastic velocity $\mathbf{u}_e$ computed by a Laplacian filter tends to centralize the position of the vertices in relation to its neighbors. This movement minimizes the amount of insertions and deletions depending on the parameters β_1 and β_2 from eq. (2).

7 NUMERICAL RESULTS

In this section, results for the static bubble and oscillating drop computations are presented in order to validate the proposed method. Additionally, results for rising bubble and bubble coalescence are presented, and they are compared with other techniques, demonstrating the advantages of this method.

7.1 Static bubble

A static bubble immersed into another fluid is simulated to verify the surface tension calculation and measure the influence of parasitic currents in the flow. This problem was simulated in a 2×2 domain discretized by four different unstructured meshes with constant grid spacing, $h = 0.2$, $h = 0.1$, $h = 0.05$ and $h = 0.025$. The validation is performed comparing the pressure jump at the interace with known analytical value, given by the Laplace formula[9] as

$$\Delta p = p_b - p_f = \sigma\kappa = \frac{\sigma}{R} \tag{19}$$

where p_b is the internal bubble pressure, p_f is the external pressure and R is the radius of the bubble. Figure 2 shows the pressure profiles in a horizontal line in the middle of the domain. The profiles obtained are flat, which illustrate the precision and the sharpness of the interface representation. Table 1 shows the number of elements, pressure jump, relative error and the magnitude of the parasitic currents for the several grid spacings.

Grid spacing	Ne	Δp	E	$\max\{\lvert u\rvert, \lvert v\rvert\}$
$h = 0.2$	248	1.97847	1.088%	1.638×10^{-4}
$h = 0.1$	946	1.99485	0.258%	2.015×10^{-5}
$h = 0.05$	3782	1.99872	0.064%	4.535×10^{-6}
$h = 0.025$	14972	1.99969	0.015%	4.252×10^{-6}

Table 1: Comparison between pressure jumps and relative errors for several grid spacings. In this table, Ne is the number of elements, Δp is the pressure jump at the interface and E is the percentage of the approximation error.

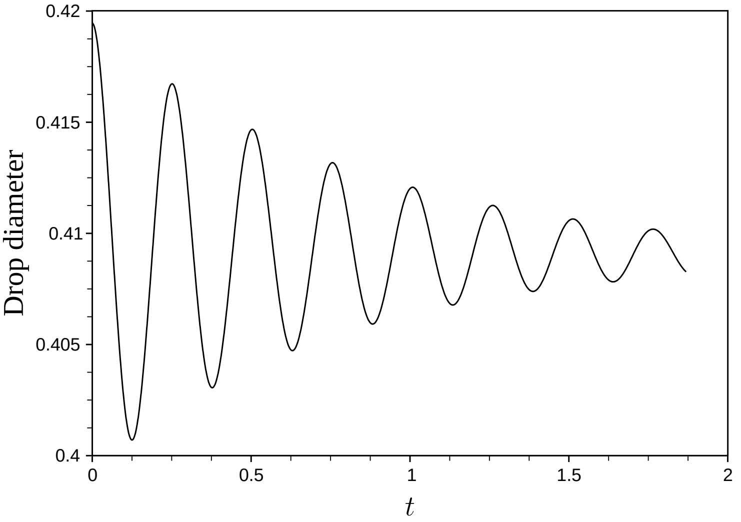

Figure 3: Oscillation of the bubble diameter, showing the decay due to viscosity. The error found in frequency of oscillation is about 9%.

The results are in good agreement with analytical values. From the results reported in table 1, it can be concluded that the method shows a second order convergence for the interface tension computation.

7.2 Oscillating drop

To verify that the transfer between the surface energy and kinetic energy is correctly accounted for, which is important when the flow is dominated by surface tension effects, we validated the method on an eliptic oscillating drop, for which an analytical solution for the oscillation frequency exists. This problem consists of simulating an elliptic drop immersed in a continuous lighter and less viscous phase without a gravity field. The drop has a small initial perturbation with respect to its equilibrium circular form and, driven by the interfacial forces, it tends to oscillate. The non-dimensional parameters chosen for the simulation are diameter $D = 0.4$, density $\rho_d = 1$ e viscosity $\mu_d = 0.01$, with the initial horizontal diameter 5% greater than the vertical diameter. The density ratio between drop and external fluid is $\rho_d/\rho_f = 20$ and viscosity ratio $\mu_d/\mu_f = 10$. This validation was simulated in an unitary domain discretized by a non-uniform mesh with $h_{min} = 0.02$ on the interface and $h_{max} = 0.05$ on the boundaries.

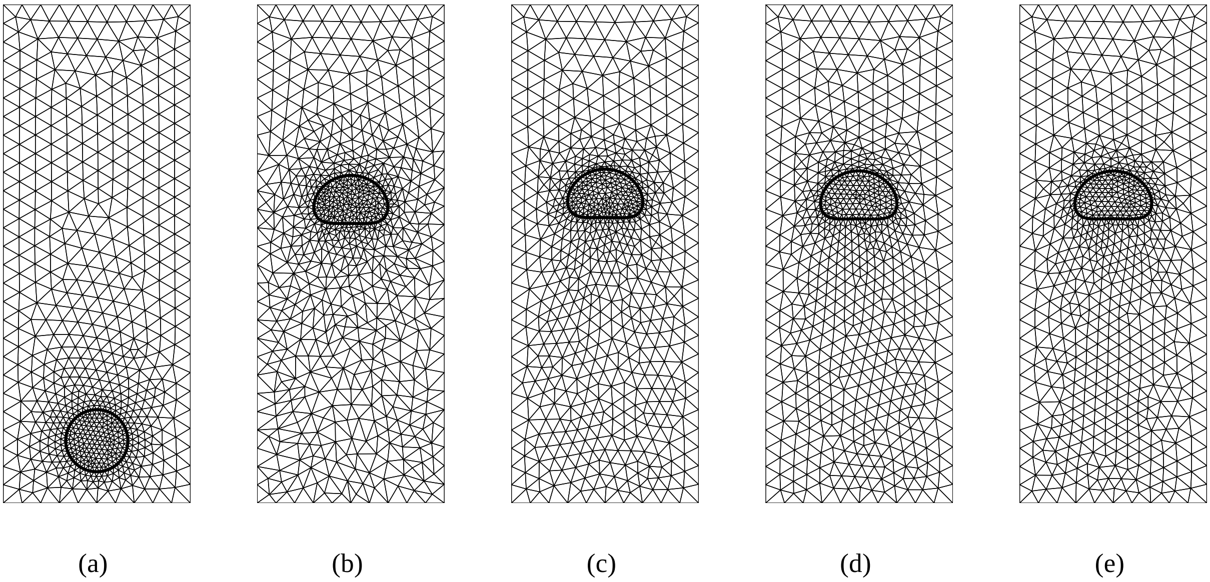

Figure 4: Initial mesh (a) and comparison between the meshes at $t = 12.5$ for the parameters: (b) $\beta_2 = 0$, (c) $\beta_2 = 0.1$, (d) $\beta_2 = 0.5$, (e) $\beta_2 = 1.0$.

The results obtained using the Lagrangian approach to move the grid points showed an error about 9% in the frequency of oscillation, which in comparable to other results in literature for equivalent resolution and domain.[2]

7.3 Rising bubble

In order to compare the influence of the elastic velocity on the results, a rising bubble test case was simulated for several parameter β_2, which determines the amount of elastic velocity present in the mesh velocity. The nondimensional parameters, taken from,[14] are $M = 0.1$, $Eo = 10$, $\rho_f/\rho_b = 100$, $\mu_f/\mu_b = 2$ and $R = 0.5$, where ρ_b, μ_b are the density and viscosity for the bubble, and ρ_f, μ_f are density and viscoty for the fluid. The mesh is initially discretized by 1306 elements with $h_{min} = 0.08$, in a 2×6 domain. The parameters for the computation of the mesh velocity are $\beta_1 = 1$ for all cases, and each case simulated with a different β_2: $\beta_2 = 0$, $\beta_2 = 0.1$, $\beta_2 = 0.5$ and $\beta_2 = 1$. Figure 4 displays the meshes obtained in each simulation.

Figure 5 shows a comparison of the Reynolds number and mass conservation between the simulated cases. In these figures we can observe an oscillation in the rising velocity of the bubble when increasing the parameter β_2, while the rising velocity for $\beta_2 = 0$ is smooth. This oscilation can be explained by the amount of mesh changes done at once in these simulations. The higher the parameter β_2, the harder the mesh is to move, and as the front is moving freely across the domain, the mesh has to adapt killing and creating many cells at once, which generates the rising velocity oscillations.

On the other hand, the mass conservation reported on figure 5 clearly shows that the higher

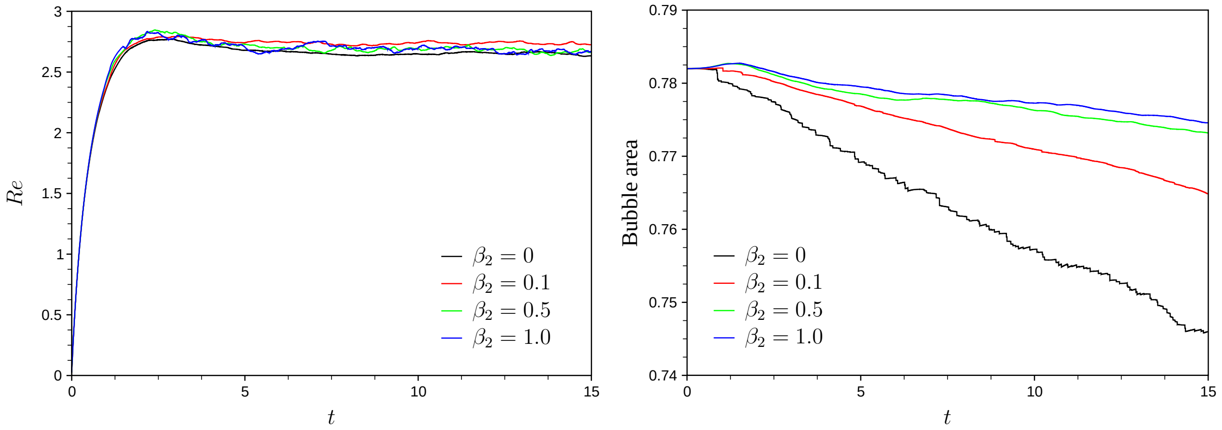

Figure 5: Bubble Reynolds number and mass conservation for the simulated cases.

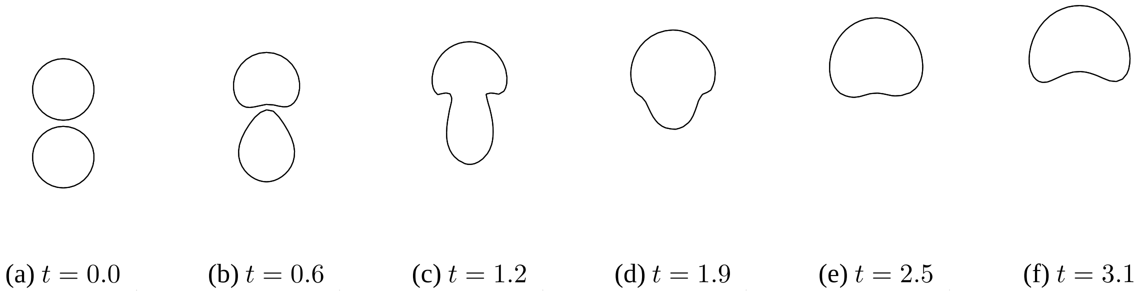

Figure 6: Colaescence of two bubbles of the same fluid rising in an quiescent fluid.

β_2, the better the mass conservation. While in a ALE fashion, higher values of β_2 results in insertions and deletions of many vertices at once, the number of mesh changes is far less than in a Lagrangian fashion. In simulations with $\beta_2 = 0$, the mesh has to be adapted more often, mainly over the interface, which causes the higher loss of mass.

7.4 Bubble coalescence

To show that the method can easily deal with topological changes at the interface, we simulated the coalescence of two rising bubbles. The parameters for this problem are the same as the previous simulation, with two bubbles of the same fluid and same size $R = 0.5$ being released at a distance of $0.1D$. As the two interfaces get closer to each other, at a distance of one element, coalescence takes place. We simply check for the existence of elements in the surrounding fluid that have three interface vertices, and change the material of this element if this will not result in a singularity. Reflagging of vertices that no longer belong to the interface is also performed to avoid singularities. Figure 6 illustrates the coalescence of two bubbles. This result shows good qualitative agreement with other results reported in the literature.[8, 13]

8 CONCLUSION

This paper presents a method to simulate incompressible multi-fluid flows in which the mesh moves in an ALE fashion. The interface between fluids was represented by vertices and edges of the triangulation plus a level-set of a Heaviside function, which is also used to compute the interfacial force distribution. The discussion focuses the computation of the mesh velocity, as well as implementations on moving finite element unstructured meshes, using remeshing procedures to avoid bad elements. The conservation equations are solved by a projection method based on approximated block *LU* decomposition of the system of equations, to decouple the acceleration and pressure.

Validations were performed for the approximation of the interfacial force, by the simulation of static bubble and oscillating drop. The static bubble simulation showed that the parasitic currents are very small, as expected, and a convergence study reveals second order convergence for the curvature calculation. The oscillating drop simulation also showed good agreement for the oscillation frequency, with error comparable to other works in literature.

Results for the rising bubble are compared for several mesh velocities, showing the better mass conservation properties obtained with the introduction of an elastic velocity, computed by a Laplacian filter. On the other hand, if this smoothing is too strong, oscillations in the bubble velocity can appear, effect that can be easily controlled by the parameter β_2. Additionally, a bubble coalescence simulation was performed in order to demonstrate the capabilities of the proposed method to deal with topologic changes at the interface.

ACKNOWLEDGEMENTS

This work was partially developed at GESAR/UERJ (Center for Environmental Studies and Simulations for Reservoirs). The authors thank brazilian agency FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo, grant number 01/12623-5), CNPq (grant numbe 302633/2003-0), and PROCIENCIA/UERJ/FAPERJ (Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro), and Furnas Centrais Elétricas S.A., for the financial support.

REFERENCES

[1] S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *Journal of Computational Physics*, **100**, 25–37 (1992).

[2] A. Esmaeeli and G. Tryggvason. Direct numerical simulations of bubbly flows. part 1. low reynolds number arrays. *Journal of Fluid Mechanics*, **337**, 313–345 (1998).

[3] A. Esmaeeli and G. Tryggvason. Direct numerical simulations of bubbly flows. part 2. moderate reynolds number arrays. *Journal of Fluid Mechanics*, **385**, 325–358 (1999).

[4] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Janz. A front-tracking method for the computations of multiphase flow. *Journal of Computational Physics*, **169**, 708–759 (2001).

[5] S. Osher and J. A. Sethian. Fronts propagation with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, **79**, 12–

49 (1988).

[6] Chen T., Minev P. D., and Nandakumar K. A projection method for incompressible multiphase flow using adaptive eulerian grid. *International Journal for Numerical Methods in Fluids*, **45**, 1–19 (2004).

[7] J. B. Perot and R. Nallapati. A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows. *Journal of Computational Physics*, **184**(1), 192–214 (2003).

[8] F. S. Sousa and N. Mangiavacchi. A lagrangian level-set approach for the simulation of incompressible two-fuid flows. *International Journal for Numerical Methods in Fluids*, **47**, 1393–1401 (2004).

[9] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modeling surface tension. *Journal of Computational Physics*, **100**, 335–354 (1992).

[10] J. B. Perot. An analysis of the fractional step method. *Journal of Computational Physics*, **108**, 51–58 (1993).

[11] W. Chang, F. Giraldo, and B. Perot. Analysis of an exact fractional step method. *Journal of Computational Physics*, **180**, 183–199 (2002).

[12] M. J. Lee, B. D. Oh, and Y. B. Kim. Canonical fractional-step methods and consistent boundary conditions for the incompressible navier-stokes equations. *Journal of Computational Physics*, **168**, 73–100 (2001).

[13] F. S. Sousa, N. Mangiavacchi, L. G. Nonato, A. C. Castelo, M. F. Tomé, 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. *Journal of Computational Physics*, **198**, 469–499 (2004).

[14] A.-K. Tornberg. *Interface tracking methods with applications to multiphase flows*. PhD thesis, Royal Institute of Technology (KTH), Stockholm, Sweden, (2000).