A 3D ALE Finite Element Method for Two-Phase Flows with Phase Change

THÈSE N^{O} 5426 (2012)

PRÉSENTÉE LE 24 JUILLET 2012
À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DE TRANSFERT DE CHALEUR ET DE MASSE
PROGRAMME DOCTORAL EN MÉCANIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Gustavo RABELLO DOS ANJOS

acceptée sur proposition du jury:

Prof. M. Parlange, président du jury
Prof. J. R. Thome, Dr N. Borhani, directeurs de thèse
Prof. F. Gallaire, rapporteur
Prof. N. Mangiavacchi, rapporteur
Prof. I. Zun, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2012

to my wife Michelle…

Acknowledgements

This work has been carried out at the Laboratory of Heat and Mass Transfer (LTCM), Swiss Federal Institute of Technology in Lausanne and funded by the Nano-Tera RTD project CMOSAIC (ref. 123618), financed by the Swiss Confederation and scientifically evaluated by SNSF.

Many people contributed or helped in some way the completion of this doctoral thesis. Firstly, I would like to express my deepest gratitude to my advisor, Prof. John R. Thome, for his strong encouragement, support and fruitful conversation over the past 4 years. Also, I would like to thank my co-director Dr. Navid Borhani who always believed in the value of this research.

I would like to thank Prof. Norberto Mangiavacchi for his advices during the development of the numerical code and the countless discussions we had. I am equally grateful to Prof. Marc Parlange, Prof. François Gallaire and Prof. Iztok Zŭn for their valuable time in reading this manuscript, together with their constructive remarks and suggestions.

It is my pleasure to thank all my colleagues of LTCM, with whom I spent many joyful days, but a special thank goes to those that in some how contributed to the completion of this thesis and made my days more pleasant, namely Sylwia Szczukiewicz, Duan Wu, Sepideh Khodaparast, Nicolas Lamaison, Jackson B. Marcinichen, Jean-Baptiste (LENI), Jeff Ong, Mathieu Habert, Nathalie Matthey-de-l'Endroit, Cécile Tavernay and Laurent Chevalley. I am heartily thankful to Dr. Bogdan A. Nichita and Dr. Mirco Magnini for many fruitful discussions about two-phase flows and numerical analysis as well as great friendship.

I joyfully send my regards to Prof. Jose Pontes for his support on my studies.

The words are not enough to thank my family for their encouragement and support. However, I would like to dedicate my special thank to my brother, André R. dos Anjos, and my sister-in-law, Ana Carolina C.B.R. dos Anjos, who have made possible and enjoyable my stay in Switzerland. Moreover, I would like to express my deepest feelings of thanks to my mother, Léa Rabello, for her everlasting support and my aunt, Zuleika Rabello, for her kindness.

Last, but not least, I would like to thank my wife, Michelle Bastos, for whom this thesis is dedicated, for companionship, affection and countless support during each day of my doctoral studies.

Abstract

A new numerical method is proposed to study two-phase flow and heat transfer for interlayer cooling of the new generation of multi-stacked computer chips. The fluid flow equations are developed in 3-dimensions based on the Arbitrary Lagrangian-Eulerian formulation (ALE) and the Finite Element Method (FEM), creating a new two-phase method with an improved model for the liquid-gas interface. A new adaptive mesh update procedure is also proposed for effective management of the mesh at the two-phase interface to remove, add and repair surface elements, since the computational mesh nodes move according to the flow. The Lagrangian description explicitly defines the two-phase interface position by a set of interconnected nodes which ensures a sharp representation of the boundary, including the role of the surface tension. The new methodology for computing the curvature leads to accurate results with moderate programming effort and computational cost. Static and dynamic tests have been carried out to validate the method and so far all the obtained results have compared well to analytical solutions and experimental results found in the literature, demonstrating that the new proposed methodology to simulate two-phase flows provides good accuracy to describe the interfacial forces and bubble dynamics. The new code was then used to simulate elengated bubble flows in square microchannels, being considered for two-phase interlayer cooling in future 3D-IC compute chips.

Keywords: Two-Phase flows, Surface tension, Curvature, Arbitrary Lagrangian Eulerian, Finite Element Method, Adaptive mesh refinement.

Résumé

Une nouvelle méthode numérique est proposée pour étudier les écoulement biphasiques et le transfert de chaleur lors du refroidissement "inter-couches" de la nouvelle génération de processeur "multi-stack". Les équations d'écoulement du fluide sont développées en 3-dimensions et basées sur la formulation Lagrangiene-Euleriene Arbitraire (LEA) et les Méthodes des Elements Finis (MEF). Ceci représente une nouvelle méthode de modelisation des ecoulements biphasiques comportant une approche améliorée de l'interface liquide-gaz. Une nouvelle procédure d'adaptation de maille est également proposée pour une gestion efficace de la maille à l'interface biphasique afin de supprimer, ajouter et réparer les éléments de surface, les noeuds du maillage se déplaćant avec l'écoulement. La description Lagrangiene définit explicitement la position de l'interface biphasique par un ensemble de noeuds interconnectés qui assure une représentation nette de la frontière, incluent notamment le rôle de la tension de surface. La nouvelle méthodologie de calcul de la courbure conduit à des résultats précis avec une complexité de programmation et un temps de calcul modérés. Des essais statiques et dynamiques ont été réalisés pour la validation de la méthode et à ce jour, tous les résultats obtenus sont en accord avec les solutions analytiques et les résultats expérimentaux trouvés dans la littérature. Ceci démontre que la nouvelle méthodologie de simulation des écoulements biphasiques présente une bonne précision pour décrire les forces interfaciales et la dynamique des bulles. Le nouveau code a été ensuite utilisé pour simuler les écoulements à bulles allongée dans des microcanaux carrés, représentant le futur refroidissement intermédiaire biphasique des processeurs 3D-IC.

Mot clés: Écoulement biphasique, Tension de Surface, Courbure, Lagrangiene Euleriene Arbitraire, Méthodes des Elements Finis, Adaptation de maille.

Contents

Acknowledgements **v**

Abstract (English/Français) **vii**

List of figures **xiii**

List of tables **xxiii**

Nomenclature **xxv**

1 Introduction **1**

2 Literature review **3**
- 2.1 Finite Element Method 3
- 2.2 Two-Phase Flow Methods 5
- 2.3 Adaptive Mesh Refinement 12
- 2.4 Mass Transfer Models 15
- 2.5 Conclusions 17

3 Governing Equations **19**
- 3.1 Introduction 19
- 3.2 Arbitrary Lagrangian-Eulerian description 20
- 3.3 Conservation of mass 22
- 3.4 Conservation of momentum 24
- 3.5 Conservation of energy 28
- 3.6 Non-dimensional form of the Navier-Stokes equation 30
  - 3.6.1 Mass Transfer 36
  - 3.6.2 Generic initial and boundary conditions 38

4 Finite Element Method **39**
- 4.1 Variational Formulation 39
- 4.2 The semi-discrete Galerkin Method 44
- 4.3 The semi-Lagrangian Method 50
  - 4.3.1 Semi-Lagrangian Method for the Navier-Stokes Equations 53
- 4.4 Mesh elements 54

4.4.1 Volume coordinates 55
4.4.2 3-dimensional elements 57
4.5 The Delaunay Tetrahedralization 60
4.6 Projection Method 61
4.6.1 The Projection Method 62
4.6.2 The LU-based discrete projection method 64
4.6.3 Time step restriction 67
4.6.4 Object-oriented design 68

5 The Discrete Interface **73**
5.1 Geometrical representation 73
5.2 Curvature and normal vectors in $\mathbb{R}^3$ 77
5.3 The discrete surface tension force 82

6 Adaptive mesh refinement **83**
6.1 Mesh representation 83
6.2 Volumetric nodes 84
6.3 Mesh smoothing 85
6.4 Surface remeshing 91
6.4.1 Point insertion 91
6.4.2 Point deletion 92
6.4.3 Edge contraction 96
6.4.4 Edge flipping 96
6.4.5 Volume Conservation 98

7 Validations and Results **103**
7.1 Curvature calculation 103
7.2 Static droplet 106
7.3 Sessile drop 108
7.4 Oscillating drop 110
7.5 Falling Drop in an Inert Media 113
7.6 Spherically growing bubble in a superheated liquid 114
7.7 Rising Bubble 116
7.8 Rising of Taylor Bubbles 122
7.9 Two drop collision 135
7.10 Microchannel simulations 140

8 Conclusions **153**
8.1 Further work 154

A Important theorems **155**

B TetGen - command line switches **157**

Bibliography **166**

Curriculum Vitae **167**

List of Figures

2.1 Interface representation in two-phase flows. (a) In the *Eulerian* approach, the interface between the fluids is located somewhere in between the computational elements. On the other hand (b) the *Lagrangian* description represents the interface by computational objects, such as nodes, segments and elements, thus achieving a sharp interface. 6

2.2 Interface representation in the VOF method. (a) The interface is defined implicitly and it is located somewhere in the computational grid. (b) The VOF function is assembled by considering the volume occupied by each phase at each computational element. 7

2.3 Representation of the distance and level-set functions. The interface between the fluids is located in the middle of the domain. (a) The distance function is created by computing the distance of each mesh node $\mathbf{x}$ to its closest interface node $\mathbf{x}_I$. (b) The level-set functions is a signed distance function with positive values in one phase and negative values in the other phase. 8

2.4 The Heaviside function is defined as 1 in one fluid and 0 in the other. In the ALE-FE method, an average value of 0.5 is assigned to the nodes belonging the interface. (a) 2-dimensional drop immersed in another fluid. (b) 3-dimensional drops immersed in another fluid. The left drop is sliced to show the nodes belonging the phase ϕ_1. 11

2.5 Examples of Heaviside functions applied to Level-Set and Front-Tracking codes. (a) Heaviside with no artificial numerical treatment. (b) Smoothed Heaviside function used in two known references ([86] and [93]). 13

3.1 One-dimensional examples of the (a) *Lagrangian* description, in which the mesh nodes move according to the flow field, (b) *Eulerian*, in which the mesh is fixed in the space and (c) ALE, in which a generalized description is achieved. 21

3.2 Material, referential and spatial configuration for the Arbitrary Lagrangian-Eulerian framework. 22

4.1 1-dimensional space scheme of the semi-Lagrangian method. The point x_d is found by integrating the mesh backward in time according to Eq. (4.71). Thus, to calculate the quantity value, an interpolation is performed considering the quantity values on the nodes x_{i-1} and x_i. 52

4.2 2-dimensional particle trajectory from the current node's position ψ_i^{n+1} to its corresponding departure point ψ_d^n. 52

4.3 Interpolation procedures in the semi-Lagrangian method. 53

4.4 Volume coordinates for the generic tetrahedron i, j, k, l where P is a point somewhere inside the tetrahedron. 55

4.5 2-dimensional representation of a (a) *Voronoi diagram* and (b) its tessellation according to the Delaunay properties. 61

4.6 Simplified UML class diagram of the code's design. The classes are projected to allow Reusability and further development. 70

5.1 Geometrical representation of the interface between the phases. (a) The interface (gray colored) is represented by a set of triangles, edges and nodes which are part of the tetrahedron mesh. (b) The fluid property ϕ, such as density or viscosity, is sharply defined in phase 1 and phase 2 with a zero thickness interface in the transition zone. 74

5.2 Density distribution in two-phase flows. Phase 1 has a density $\rho_1 = 1000$ and phase 2 has a density $\rho_2 = 1$. The interface is at $x \approx 0.5$. (a) The sharp transition is achieved by the ALE-FE method, in which no artificial smoothing is required. (b) Smoothed distribution of density commonly found in Level-Set methods (see [87] and [93]). 76

5.3 The 2-dimensional *Frenet*'s formula for mean curvature ([48]). (a) The continuous description and (b) the discrete form used in the computational grid. . . . 78

5.4 Normal vector evaluation in 2-dimensional spaces. (a) The normal vector of each edge may be found by rotating the previously calculated tangent vector by 90°. (b) The final nodal normal vector $\mathbf{n}$ is found by summing the two normal vectors n_1 and n_2. 79

5.5 Schematic representation of the curvature evaluation κ_i at a common surface node, which is calculated using geometric operations at all the triangular neighboring elements and weighted by the barycentric area (gray colored). 80

5.6 (a) An elemental force evaluation is done using the sum of the distributed forces $\mathbf{t_{1n}}d_1$ and $\mathbf{t_{2n}}d_2$. (b) Using the Stokes theorem, the elemental distributed force may be calculated orthogonalizing one of the two linearly independent vectors t_1 and t_2 to the segment d which connects two mid-edge nodes. (c) An evaluation of the node mean curvature is found by dividing the sum of the module of the calculated distributed forces ($|t_n d|$) by the sum of the barycentric areas (Eq. 5.7.) 80

5.7 Normal vector evaluation in 3-dimensional spaces. (a) The normal vector of each triangle in the surface may be found by applying the cross product of two tangent vectors which lie in the triangle plane. (b) The final nodal normal vector $\mathbf{n}_i$ is found by summing the normal vectors n_e for $e = \{1..j\}$. In the illustrated case $j = 5$. 81

6.1 Data-set representation of the meshes used in the present numerical code. (a) The surface meshes, which comprise the interface between the fluids and the domain boundary, are passed as an input parameter to the open source library *TETGEN*, which export the 3-dimensional connectivity array. (b) The volumetric mesh is than used to discretize the two-phase flow equations. 84

6.2 Solutions of the Helmholtz's equations for different diffusive parameter k. (a) The sample was taken along the z axis, in which the bubble's location can be seen within the interval $z = \{2,4\}$. (b) The y component represents the channel cross section. In this case, the mesh is more refined close to the channel's boundaries ($y = -0.5$ and $y = 0.5$) and coarser in the middle. 86

6.3 Laplacian smoothing operation in 3-dimensional space. (a) initial point position and (b) final point position after successively smoothing steps. 87

6.4 Velocity smoothing operation in 2-dimensional spaces. Near the interface, the nodes are more influenced by the surface velocity (large arrows), while if the node is located far from the surface, the mesh velocity $\hat{\mathbf{u}}_v$ is less pronounced (small arrows). Its analogy to 3-dimension space is straightforward by considering a surface embedded in $\mathbb{R}^3$ as the interface between the fluids. 89

6.5 Normal and tangent components of the interface's velocity vector. The proposed scheme allows to remove partially or totally the tangent component of the interface's velocity $\mathbf{v}_I$ by varying the parameter γ_1. 90

6.6 Insertion of a surface node. (a) The edge $1-2$, which is longer than a fixed parameter h_{max}, is identified. (b) The new node is then added at the midpoint of the edge $1-2$. 92

6.7 Representation of the 3-dimensional triangular surface mesh. (a) The node v is added at the midpoint of the edge $1-2$ (b) The plane θ is derived by the mean of two element normal vectors which are adjacent to the edge $1-2$. The vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ are the projection of the normal vectors of nodes 1 and 2 onto the plane θ. (c) The node's new position is found by moving it from the edge $1-2$ toward the circle segment in (b), thus the curvature error in v is reduced. 93

6.8 Deletion of a surface node. (a) The edge $3-v$ is detected when its length is smaller than a reference length h_{min}. Due to the sum of neighbor edge lengths, the node v is chosen to be deleted. (b) Therefore, the empty polyhedron must be reconnected to achieve a new surface triangulation. 94

6.9 Remeshing of a surface polyhedron by successive node re-connections. (a) An edge is created by connecting the nodes 1 and 2. (b and c) The node 1 is then connected to the remaining nodes 4 and 5, thus achieving the final surface triangulation. 94

6.10 Reconstruction of the surface mesh by the "ear" technique. (a) First "ear" is achieved by connecting the nodes $1-2-3$ and forming the surface triangle. The node 2 is then deleted from the polyhedron P. (b) The new triangle is formed by connecting the nodes $3-4-5$, and thus the node 4 is eliminated. (c) Last two triangles are created from nodes $5-6-1$ (node 6 is deleted) and $1-3-5$, which are the remaining nodes of the successively deleted polyhedron. 95

6.11 Contraction of a surface edge. (a) The edge h (segment 1-2) is found to be smaller than h_{min} and (b) so it is collapsed to the midpoint of the same edge. Due to its simplicity, only triangles e_1 and e_2 are eliminated from the surface mesh and the remaining node connectivity is not affected. The new location of node 1 should respect the curvature of its neighbors as described in the insertion strategy. . . 96

6.12 Node displacement according to neighbor's curvature in the process of edge contraction. (a) The plane Ω is found using the curvature vectors of nodes 1 and 2, thus a circle equation is fitted and (b) its solution is used to displace the node and avoid losses of mass. (c) The resulting scheme of edge contraction considering the neighbor's curvature. 97

6.13 Triangular surface flipping operations: (a) The triangle aspect ratio, the curvature of neighboring nodes, and the triangle circumcenter are taken into consideration to perform the flipping from edge $1-2$ to $3-4$; (b) the flipping of edge $1-2$ cannot be assigned due to an inconsistent mesh generation. (c) The flipping operation may lead to local loss of mass and it should be treated with care. . . . 99

6.14 Volume correction in the rising bubble test case: (a) The initial volume $V \neq 0.5191$ is computed when the simulation starts, then it is compared to the current bubble's volume and corrected after a few iterations. (b) Convergence error of bubble's volume correction. 101

7.1 Comparison of a Taylor bubble and three representative shapes used to evaluate the curvature error of the proposed numerical method: (a) Taylor bubble, (b) sphere, (c) cylinder and (d) torus. The sphere with radius $R = 0.5$ has both curvatures with same value and sign. In the cylinder, part of its shape (curved) has one of the principal curvatures zero (along its height) and the other is inversely proportional to the cylinder radius $Rc = 0.5$. The torus has the principal curvatures with opposite signs in its inner part and same positive sign for the remaining shape. 105

7.2 Log scale graph showing the convergence order of the new methodology for computing the curvature of a surface. The slope found in the numerical method suggests its convergence is first order. 106

7.3 Chordal pressure jump between the phases for different surface edge lengths. The solution of the pressure field for a static droplet immersed in a low viscous fluid was interpolated in a linear uniform mesh where the non-dimensional pressure $p = 0$ corresponds to the area occupied by the gas phase and the pressure $p = 20$ stands for the area occupied by the droplet.The test was performed considering the non-dimensional radius $R = 0.5$ and $Eo = 0.2$, resulting $\Delta p = 20$. 108

7.4 Capillary pressure of a spherical droplet immersed in another fluid. The jump in pressure can be seen at the location of the interface. 109

7.5 Curvature distribution along the drop's height. The solid line was fit by the least square method and its slope gives the value of $\hat{\mathrm{Eo}} = 2$. 111

7.6 Comparison between the numerical solution of an axisymmetric sessile drop and the analytical solution of its shape derived by the Young-Laplace equation of capillarity. 111

7.7 Drop oscillation amplitude. Comparison between numerical and analytical solution for two levels of mesh refinement. The analytical period is 0.785 and the decay rate is shown by the envelope represented by the lines below and above the oscillating curve. The oscillating period in the coarse mesh was found to be 0.820 for the coarse mesh while that for the refined mesh was 0.783. 113

7.8 Inversion of the velocity direction in the z-axis. (a) The top and bottom parts are squeezing the drop, and it corresponds to $t \approx 0.0$. (b) The velocity is inverted, thus the drop is being stretched in the z-axis, $t \approx 0.4$ 114

7.9 Trajectory of the falling drop immersed in different fluids. The drop trajectory of the high viscosity fluid (black squares) tends to deviate as expected from the analytical solution while the low viscosity fluid trajectory almost matches the free fall equation. 115

7.10 Drop's fall velocity driven by gravitational effects. The effects of deacceleration are stronger when the inert media has a high viscosity compared to the analytical solution of the free fall velocity profile, while in the low viscosity fluid, the deacceleration is more pronounced, obstructing the descent of the drop. 116

7.11 Growth of a vapor bubble due to evaporation. 117

7.12 Velocity components of a vapor bubble growing due to evaporation of the superheated liquid. (a)-(c) x-component , (d)-(f) y-component and (g)-(i) z-component.118

7.13 Rising of an air bubble immersed in the aqueous sugar solution with the highest viscosity $\mu = 2.73$. (a) Bubble's shape evolution. (b) Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional. 119

7.14 Rising of an air bubble immersed in the aqueous sugar solution with moderate viscosity $\mu = 1.28$. (a) Bubble's shape evolution. (b) Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional. 120

7.15 Rising of an air bubble immersed in the aqueous sugar solution with viscosity $\mu = 0.54$. (a) Bubble's shape evolution. (b)Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional. 120

7.16 Rising of an air bubble immersed in the aqueous sugar solution with viscosity $\mu = 0.28$. (a) Bubble's shape evolution. (b)Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional. 121

7.17 Incomplete rising of an air bubble immersed in the aqueous sugar solution with the least viscosity $\mu = 0.13$. (a) Bubble's shape evolution. (b)Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional. 122

7.18 Flow pattern map of rising bubble in cylindrical vertical tubes [97]. 7 regions were identified to characterize the dependency of the velocity to viscous, interfacial and inertial forces. The horizonta axis stands for the *Eötvös* number and the vertical axis to the *Morton* number. The curved lines are dimensionless velocity, which is equivalet to the *Froude* number. 124

7.19 Schematical representation of the moving frame technique, which uses the principles of relative velocity to shorten the numerical domain. (a) The rising bubble moves with velocity V_b and the camera tracks its motion with the same velocity. (b) The boundary conditions applied to simulate the same condition illustrated in (b) requires that the walls move downward with velocity V_b. 125

7.20 Tetrahedron mesh used to simulate the rising Taylor bubble. The boundary mesh is more refined close to the bubble to capture the mechanisms of the thin liquid film. Above the bubble, the mesh is less refined and behind the the bubble a fine mesh is used to capture the bubble's wake. 126

7.21 Bubble shape evolution with time for an air bubble in a sugar syrup solution with dimensionless numbers $Mo = 10^4$, $Eo = 400$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 21.19$. 127

7.22 Rising of an air Taylor bubble immersed in the sugar syrup solution with dimensionless numbers set to $Mo = 10^4$ and $Eo = 400$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional. 127

7.23 Bubble shape evolution with time for an air bubble in a sugar syrup solution with dimensionless numbers $Mo = 10^6$, $Eo = 3$. In such a condition, the bubble does not rise due to the stronger viscous force compared to the gravitational force. (a) Initial bubble shape with $t = 0$. (b-e) Bubble shape transient solution. 128

7.24 Rising of an air Taylor bubble immersed in the second sugar syrup solution with dimensionless numbers set to $Mo = 10^6$ and $Eo = 3$. According to [97], the terminal bubble's velocity is zero, thus the bubble does not rise. The numerical simulation shows that the bubble does not rise, however a small residual is found in the rising velocity w. Velocity and time are non-dimensional. 129

7.25 Bubble shape evolution with time for an air bubble in a sucrose solution with dimensionless numbers $Mo = 10^{-7}$, $Eo = 40$. The adaptive mesh refinement proposed in this work captures accurately the strong shape distortion produced by the high ascension velocity. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape change during transient solution. (e) Terminal bubble shape with $t = 7.41$. 129

7.26 Rising of an air Taylor bubble immersed in a sucrose solution with dimensionless numbers set to $Mo = 10^{-7}$ and $Eo = 40$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional. 130

7.27 Bubble shape evolution with time for an air bubble in a sucrose solution with dimensionless numbers $Mo = 10^{-7}$, $Eo = 100$. The high ascension velocity and the strong deformation of the surface mesh shut the simulation down. Mesh analysis suggest that a specific surface mesh refinement is required to handle such a deformation. (a) Initial bubble shape with $t = 0$. (b-e) Bubble shape during transient solution. 131

7.28 Rising of an air Taylor bubble immersed in a glycerol solution with dimensionless numbers set to $Mo = 10^{-7}$ and $Eo = 100$. The sharp edged produced by the high ascension velocity is not handled by the remeshing process. Mesh analysis suggested that a different edge length is required to model high curvature shapes, which is the case of the bottom part of the bubble. Velocity and time are non-dimensional. 132

7.29 Bubble shape evolution with time for an air bubble in a glycerol solution with dimensionless numbers $Mo = 10^{-2}$, $Eo = 100$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 6.00$. 132

7.30 Rising of an air Taylor bubble immersed in a glycerol solution with dimensionless numbers set to $Mo = 10^{-2}$ and $Eo = 100$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional. 133

7.31 Bubble shape evolution with time for an air bubble in a glycerol solution with dimensionless numbers $Mo = 10^{-1}$, $Eo = 40$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 6.00$. 133

7.32 Rising of an air Taylor bubble immersed in a glycerol solution with dimensionless numbers set to $Mo = 10^{-2}$ and $Eo = 100$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional. 134

7.33 2-dimensional bubble shape evolution with time for an air bubble in a glycerol solution with dimensionless numbers $Mo = 10^{-1}$, $Eo = 40$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape during transient solution. (e) Terminal bubble shape at $t = 7.62$. 135

7.34 Qualitative comparison of bubble's terminal shape in (a) 2-dimensional and (b) 3-dimensional simulations in a glycerol solution. 136

7.35 Boundary conditions (b.c.) applied to the two drop collision simulations. (a) Background flow characterized by stagnation-point with constant strain conditions. (b) Planar case. (c) Axisymmetric case. 138

7.36 Time evolution of the collision of two equal-sized drops in the planar flow imposed by Eq. (7.15). (a-d) Dimensionless parameters are $We = 2$ and $Re = 2$. (e-h) Dimensionless parameters are $We = 2$ and $Re = 20$. (i-l) Dimensionless parameters are $We = 2$ and $Re = 200$. Time is non-dimensional. 139

7.37 Time evolution of the collision of two equal-sized drops in the axisymmetric flow imposed by Eq. (7.16). (a-d) Dimensionless parameters are $We = 2$ and $Re = 2$. (e-h) Dimensionless parameters are $We = 2$ and $Re = 20$. (i-l) Dimensionless parameters are $We = 2$ and $Re = 200$. Time is non-dimensional. 141

7.38 Film thickness δ of (a) planar case, where the z component of the velocity is not considered on the boundary conditions and (b) axisymmetric case 142

7.39 Test section of the microchannel boiling facility at the Heat and Mass Transfer Laboratory. 67 microchannes with square cross-section are distributed side-by-side and placed on the top of the micro processor. The square channel length is 100μm. Figure extracted from [57]. 143

7.40 Numerical domain used to simulate the microchannel boiling facility's microchannel. (a) The side view of the domain shows the mesh distribution, where a denser cluster of nodes are located near the bubble region. (b) The square microchannel cross-section. 144

7.41 Bubble's shape evolution with time for a single vapor bubble. The working fluid is R236fa, whose details can be found in the Table (7.6) The fluid is entering at the left of the domain and exiting at the right. A transient stage occurs in (a-b), followed by a nearly stable bubble shape in (c-e). 145

7.42 Bubble's shape evolution with time for a single vapor bubble. The working fluid is the recent environmentally friendly refrigerant R1234ze, whose details can be found in Table (7.6). (a-c) Transient stage. (d) initial formation of the surface waves in the tail of the vapor bubble. (d) Stable bubble shape. 146

7.43 Bubble's shape evolution with time for two vapor bubbles. The working fluid is refrigerant R1234ze, whose details can be found in Table (7.6). The fluid is entering at the left of the domain and exiting at the right. (a-c) Transient stage. (d) initial formation of the surface waves in the tail of the vapor bubble. (d) Stable bubble shape. 147

7.44 Transient solution of the liquid film thickness computed for the two vapor bubbles of the refrigerant R1234ze. 148

7.45 Transient solution of two-phase flow boiling of refrigerant R1234ze. A constant heat flux $\dot{q}$ is applied in the bottom part of the domain. The fluid is entering at the left of the domain and exiting at the right. Time and temperature are non-dimensional. 149
7.46 2-dimensional bubble shape evolution with time for a vapor bubble of refrigerant R1234ze. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 7.62$. 150
7.47 Qualitative comparison of bubble's terminal shape in (a) 2-dimensional and (b) 3-dimensional simulations of refrigerant R1234ze. 151

List of Tables

7.1 Standard deviation (SD) and error of curvature (κ) for different geometries. . . 104
7.2 Comparison between edge length size, pressure distribution and spurious currents.107
7.3 Fluid properties 115
7.4 Fluid properties 123
7.5 Dimensionless liquid film thickness 131
7.6 Fluid properties 144

Nomenclature

Roman Letters

A_j^i	the barycentric area	$[m^2]$
da	infinitesimal area element	$[m^2]$
N	Archimedes number	[-]
$\mathbf{c}$	ALE velocity	[m/s]
c_p	specific heat	[J/kgK]
Ca	Capillary number	[-]
$\mathbf{D}$	strain tensor	[-]
D	bubble's diameter	[m]
d	surface segment	[-]
dt	time step	[s]
e_i^j	set of surface triangles	[-]
e_{ij}	the distance between the node and each neighbor	[m]
Eo	Eötvös number	[-]
$\mathbf{f}$	surface tension force	[N]
$\mathbf{f}$	surface tension volume force	$[kg/m^3]$
Fr	Froude number	[-]
$\mathbf{g}$	gravity	$[m/s^2]$
$\mathbf{G}$	discrete gradient matrix	[-]

g	referential gravity	$[m/s^2]$
H	Heaviside function	[-]
H	heaviside function	[-]
h	edge length	[m]
h_b	initial edge length distribution	[m]
H_λ	discrete Heaviside vector	[-]
h_{fg}	latent heat	[J/kg]
h_{max}	maximum accetable mesh edge length	[m]
h_{min}	minimum accetable mesh edge length	[m]
$\mathbf{I}$	identity matrix	[-]
i	number of nodes	[-]
$\mathbf{J}$	diffusive flux	$[°C/m^2]$
j	number of triangles	[-]
k	diffusive parameter	[-]
k	material’s thermal conductivity	[W/mK]
k_∞	thermal conductivity	[W/mK]
L	characteristic length	[-]
$\dot{m}_I$	mass transfer rate at the phase boundary	[-]
$\mathbf{M}$	mass matrix	[-]
dm	infinitesimal mass	[kg]
M	Morton number	[-]
m	number of neighbor elements of i	[-]
$\mathbf{n}$	surface unity outward normal vector	[-]
N_1	set of 1-ring neighbors	[-]
n_i	be the set of nodes	[-]
p	pressure	[Pa]
p_{sys}	system pressure	[Pa]

Pe	Peclet number	[-]
Pr	Prandtl number	[-]
$\dot{q}$	heat flux	$[W/m^2]$
$\mathbb{R}^3$	3-dimensional space set	[-]
r	circunsference radius	[m]
Re	Reynolds number	[-]
s	arc length	[m]
$\mathbf{t}$	surface tangent vector	[-]
T	temperature or any scalar per volume unit	[K]
t	time	[s]
T_I	temperature of the interface between the phases	[K]
t_n	normal vector to the segment d	[-]
T_s	saturation temperature	[K]
T_w	the temperature in the wall	[K]
$\mathbf{u}$	velocity x-direction	[m/s]
$\hat{\mathbf{v}}$	mesh velocity	[m/s]
$\hat{\mathbf{v}}_e$	mesh velocity	[m/s]
$\hat{\mathbf{v}}_{v_i}$	mesh velocity	[m/s]
$\mathbf{v}$	fluid velocity	[m/s]
$\mathbf{v}$	velocity y-direction	[m/s]
$\mathbf{v}_e$	smooting velocity	[m/s]
$\mathbf{v}_I$	normal interface velocity	[m/s]
$\mathbf{v}_l$	liquid velocity	[m/s]
$\mathbf{v}_t$	tangent velocity	[m/s]
$\mathbf{v}_v$	smoothing velocity	[m/s]
$\mathbf{v}_v$	vapor velocity	[m/s]
$\mathbf{v}_{I_n}$	interface's normal velocity	[m/s]

$\mathbf{v}_{I_n}$	surface normal velocity	[m/s]
$\mathbf{v}_{I_t}$	interface's tangent velocity	[m/s]
$\mathbf{v}_{I_t}$	surface tangential velocity	[m/s]
$\mathbf{v}_{inflow}$	inflow velocity	[m/s]
dV	infinitesimal volume element	$[m^3]$
V	fixed volume element	$[m^3]$
v_x	velocity component in x direction	[m/s]
v_y	velocity component in y direction	[m/s]
v_z	velocity component in z direction	[m/s]
(We)	Weber number	[-]
$\mathbf{w}$	velocity z-direction	[m/s]
w_{ij}	weight	[-]
We	Weber number	[-]
$\hat{\mathbf{x}}$	node position according to mesh velocity $\hat{\mathbf{v}}$	[-]
$\tilde{\mathbf{X}}$	referential domain	[-]
$\mathbf{X}$	material domain	[-]
$\mathbf{x}$	spatial domain	[-]
x	coordinate	[-]
(x_c, y_c)	circumference center	[m]
y	coordinate	[-]
Z	constant which modifies the latent heat h_{fg}	[-]
z	coordinate	[-]

Greek Letters

β	average density constant	[-]
β_1	volumetric mesh parameter	[-]
β_2	volumetric mesh parameter	[-]
β_3	volumetric mesh parameter	[-]

δ	Dirac delta function with support on the interface	[-]
δ	Dirac delta function	[-]
ϕ	generic fluid property	[-]
$\dot{\gamma}(\mathbf{v})$	rate-of-strain tensor	[-]
γ_1	surface mesh parameter	[-]
γ_2	surface mesh parameter	[-]
κ	curvature	[1/m]
κ	the curvature	[1/m]
κ_{max}	maximum curvature	[1/m]
λ	volumetric viscosity	[-]
μ	dynamic viscosity	[Pas]
$\nabla\cdot$	divergent operator	[-]
∇	continuous gradient operator	[-]
∇	gradient operator	[-]
∇^2	continuous Laplacian operator	[-]
Ω_s	embedded surface in $\mathbb{R}^3$	$[-]$
ρ	fluid density	$[kg/m$^3]
ρ	specific weight	$[kg/m^3]$
Δs	discrete arc length	[m]
∂s	continous infinitesimal arc length	[m]
Σ	discrete diagonal force $\sigma\kappa$ vector	[-]
σ	stress tensor acting at each area element	[-]
σ	surface tension coefficient	[N/m]
σ_0	referential surface tension	[N/m]
τ	deviatoric stress tensor	[-]
θ	angle between planes	[^∘]

1 Introduction

Today, most of the cooling devices found in personal computers and datacenters use either single-phase air or water cooling systems. With microprocessor performance increasing exponentially, an efficent and better way to cool and decrease the computer chip temperature is of utmost importance. Furthermore, a substantial increase of the number of chips per motherboard plans to go to multi-layer stacks of chips with internal cooling channels, since higher computational resources are continouasly required. It is known that the heat exchange of two-phase flow systems are much higher than those using single-phase flow, mainly due to the nature of the thermal behavior of each phase in presence of an interface layer separting both fluids. Therefore, a new cooling technique is proposed to maintain simultansly the temperature of two or more stacked microprocessors, within an optimal working range, by flowing and evaporating two-phase environmentally friendly refrigerants in-between. These operating fluids are responsible for removing the excessive heat produced by the processors, however the cooling channels are limited to the order of 100 microns size.

Despite the available cutting-edge experimental techniques, a deeper insight into the microscale flow field is necessary. However, to access such a small length scale accurately, different techniques are required. In this context, numerical analysis has become an useful tool to simulate the mechanisms of two-phase flows, due to the fast growth of computer resources and the reduction of cost compared to those of experimental facilities. In fact, the modeling of such conditions is not an easy task due to the complexity of the non-linear set of equations that govern the flow field. Moreover, the characterization of surface tension forces and the interfacial deformation between the vapor and liquid phases adds another level of complexity, all of which require significant efforts to resolve in two-phase flow simulations.

This work is part of the larger multi-disciplinary multi-laboratory Nano-Tera/CMOSAIC project which aims to study and design microscale two-phase interlayer cooling systems for the next generation of 3-dimensional stacked microprocessors within this framework. This present thesis proposes a new “one-fluid” moving mesh methodology, using the Arbitrary Lagrangian-Eulerian description and the Finite Element method, to simulate 3-dimensional two-phase flows in macro and micro scales with phase change as an extension of the single-phase code developed previously at the Metallurgical and Materials Engineering Department of the Federal University of Rio de Janeiro (UFRJ-COPPE) and in the Group of Environmental Studies for Water Reservoires of the State University of Rio de Janeiro (UERJ-GESAR) ([4]).

The 3-dimensional Navier-Stokes and energy equations are discretized over a unstructured tetrahedral mesh through the Finite Element method. The *Mini* element is employed to fulfill the stability requirements of the *LBB* condition and the semi-Lagrangian method is used to model the convective terms, allowing large time steps and producing symmetric positive-definid matrices, whose solution is efficiently calculated. The proper description of the surface tension forces is a key aspect to sucessfully model of the two-phase flows. In this work, the interface between the phases is represented by an unstructured 3-dimensional surface, whose description is based on a set of geometrical objects such as points, edges and triangular faces. In this way, the mean curvature, used to compute the surface tension, can be calculated directly on the computational nodes by geometric functions, i.e., normal and tangent vectors based on *Frenet*'s Formula [48]. With such an approach, a sharp interface between the phases is also achieved due to its element set representation. Moreover, due to the discretization method, the thickness of the interface in the transition area of fluid properties, namely viscosity and density, is kept sharp, thus not requiring a particular function to deal with numerical instabilities in the two-phase interface. Due to the adopted description of the fluid flow equations, no velocity restriction is imposed. However, to describe accurately fully turbulent flow regimes, the mesh size should be proportional to the minimum present length scale.

The new methodology proposed in this thesis is detailed in the following chapters and several test cases are used to evaluate the accuracy of the computed surface tension force. Finally, the method is compared to some representative experimental results. The contents of this Ph.D. thesis are organized into the following chapters:

- Chapter 1: Introdution
- Chapter 2: Literature Review
- Chapter 3: Governing Equations
- Chapter 4: Finite Element Method
- Chapter 5: Interface Description
- Chapter 6: Adaptive Mesh Refinement
- Chapter 7: Results
- Chapter 8: Conclusions

2 Literature review

In this chapter, the available literature is reviewed and discussed, focusing on the methodology adopted in the present work, namely Finite Element Method, two-phase flow methods, surface tension models, adaptive mesh refinement schemes and mass transfer models.

2.1 Finite Element Method

The Finite Element method has been used, since the 1950's, in solid mechanics to solve problems in which the available standard methods were not capable of handling. Only later on, in the 1970's, the Finite Element method began to be used to discretized the governing equations in fluid dynamics, mostly due to the consolidation of the Galerkin method for the diffusion equations. The relatively late start in fluid dynamics is mainly attributed to the strong velocity-pressure coupling and convective terms found in the Navier-Stokes equations, in which the later can not be written in terms of a linear combination of independent components. Thus, the non-linearity produces non-symmetric operators whose solution is not trivial. Moreover, the greater is the *Reynolds* number Re, the larger is the influence of the convective term in the equations. For instance, in large channel flows, if Re is greater than 2000, the flow regime is characterized by its non-linearity, namely turbulence. Another numerical obstacle is the incompressibility condition, which is found in many practical problems. Such a condition imposes that the velocity field must have zero-divergence, thus the pressure field can not be coupled to any other quantity. However, many authors have started to develop important tools to investigate these problems and different approaches are found in the basic literature.

The pioneering work has been performed by [104] in which the Finite Element method was formalized for solving common field problems. A triangular mesh was set up to investigate boundary value problems and, despite the precarious computational resources, many engineering problems could be treated numerically. This work brought to the numerical research field great capabilities that were further developed by many other authors in the fluids area. Nowadays, the Finite Element method is widely used due to its enormous flexibility to solve problems in complex geometries and its fundamental mathematical framework, which allows real problems to be modeled in several different ways.

An algorithm for the solution of purely diffusion or diffusion-convective is presented in [73]. The mixed characteristic-Finite Element method is developed to derive first and second order accurate conservative upwind schemes. The algorithm can be used in the Navier-Stokes formulation; however the numerical implementation requires a quadrature formula to assemble the right hand side vectors, thus increasing the complexity of the proposed algorithm.

In [42] and [43], a new Petrov-Galerkin formulation for Stokes problems was derived. The strong *Babuska-Brezzi* condition was "circumvented" by introducing an artificial stabilization process which allowed the use of elements with the same order to discretize fluid flow equations, therefore reducing considerably the size of the final linear system and, consequently, the time spent to solve it. However, its implementation is not straightforward and the required assembling time may be larger than those found in the standard methods. While this Finite Element formulation is proposed to solve the Stokes problems, it can be successfully applied in the Navier-Stokes equations to avoid space restrictions commonly found in velocity-pressure coupling problems.

The Taylor-Galerkin method for Finite Element schemes was proposed and discussed in [26]. The method is described for scalar convection equations in one or more space dimensions, which produces accurate temporal differencing by using Taylor series expansions. In Taylor-Galerkin methods, the time discretization plays an important role; therefore the time-stepping is chosen so that the stabilization occurs in a natural way under certain time step restriction. The method is compared to the Galerkin and Petrov-Galerkin methods and the conclusion is that the new methodology produces high phase-accuracy with very low numerical diffusion.

In [53], proposed a two-step explicit Finite Element scheme was proposed to obtain time-accurate solutions for compressible Euler equations. Additionally, an adaptive mesh refinement scheme was proposed to rearrange the elements according to a required accuracy for compressible flows. The computational domain is discretized by tetrahedron elements in an uniform grid and common benchmarks are performed to evaluate the accuracy of the proposed methodology.

In [68] and [9], they studied a discontinuous Galerkin formulation applied to the Finite Element method. The method proposed was able to solve purely diffusion and convection-diffusion problems with different levels of mesh refinement. They performed several stability analysis studies and compared results with the standard continuous Galerkin method, thus demonstrating quantitatively and qualitatively the superiority of the proposed discontinuous formulation against the continuous case.

More recently, [4] developed a 3-dimensional scheme of the Navier-Stokes equations using a mixed formulation of bubble tetrahedron elements, the semi-Lagrangian method and the projection method, thus fulfilling the *Babuska-Brezzi* condition and generating a symmetric positive-definite matrix, whose solution may be more rapidly computed. The scheme was shown to be numerically stable for a large range of Reynolds numbers, while however the linear interpolation, present in the semi-Lagrangian method, may lead to excessive undesired numerical diffusion, and thus a high interpolation order is required.

Among these works, the investments in numerical two-phase flow and Finite Element method are growing exponentially. Mainly due to the high mathematical flexibility of the Finite Element method itself, but also due to its accuracy to model two-phase flow phenomena. In [89], they used the Finite Element method to model a class of unsteady moving boundary problems such as free surface flows, two-fluid interfaces, fluid-object and fluid-structure interactions and moving mechanical components using interface-tracking methods and interface-capturing methods to model the interface boundaries. Moreover, 3-dimensional complex geometries were tested and benchmarked to validate the proposed methodology.

The Finite Element method has been used to simulate free-surface flow problems with dynamic contact lines in [54]. The Arbitrary Lagrangian-Eulerian description was employed to describe the solution of the 3-dimensional equations. The interface was modeled by a triangular surface mesh and the domain mesh was successfully discretized by the quadratic tetrahedral Taylor-Hood element. Additionally, [13] investigated the same problem of free surface flows, but employing the *MINI* tetrahedron element, which reduces the number of unknowns in the final set of linear equations. Tests and error analysis were conducted, showing that the proposed methodology is suitable for modeling moving boundary problems. However, the dynamics of an additional phase were not taken into account in both works.

2.2 Two-Phase Flow Methods

In two-phase flows, discrete interface modeling is the key factor to achieve accuracy and precision. In the literature, two different approaches are commonly found, namely *Eulerian* and *Lagrangian* interface representations. These nomenclatures are strongly connected to the definition of the fluid flow motion. Figure (2.1) depicts a 2-dimensional schematic representation of the *Eulerian* and *Lagrangian* description of the interface between fluids. In the *Eulerian* formulation, also called *interface capturing*, the interface is not explicitly described, but instead is defined by special color functions which are advected by an additional hyperbolic equation. Such an equation is a common source of numerical diffusion due to its discretization and requires special attention by the computational scientist. Bubble breakups and coalescences are easily modeled by this approach. However, interface capturing requires more computational nodes to describe the scales present in the physical problem. For instance, the most widely used methods are the *Volume of Fluid* and *Level-Set* methods. They are also known as VOF and LS, respectively.

On the other hand, in the *Lagrangian* approach, the interface between the phases is represented by a set of geometric objects such as nodes, segments and faces, which may be, or may not be part of the domain mesh. Figure (2.1b) illustrates a case in which the interface is

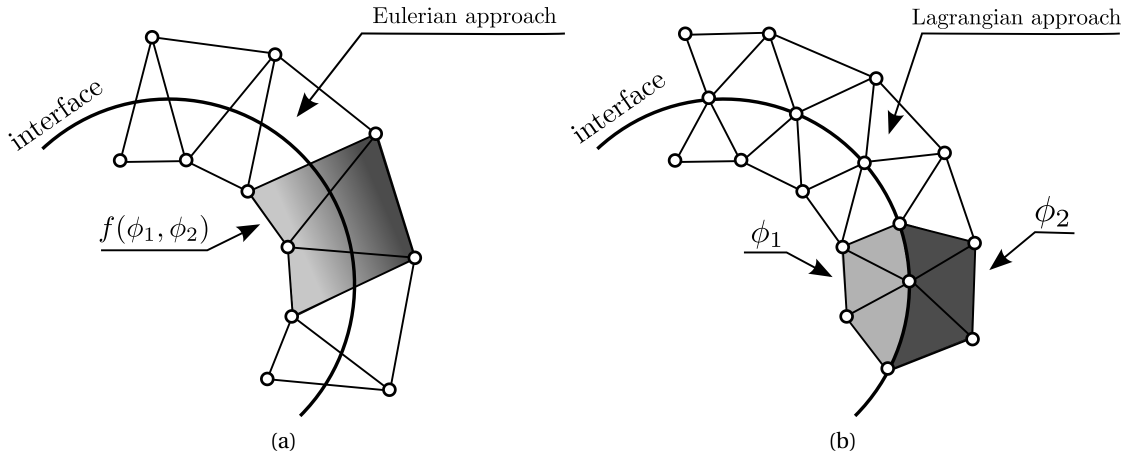


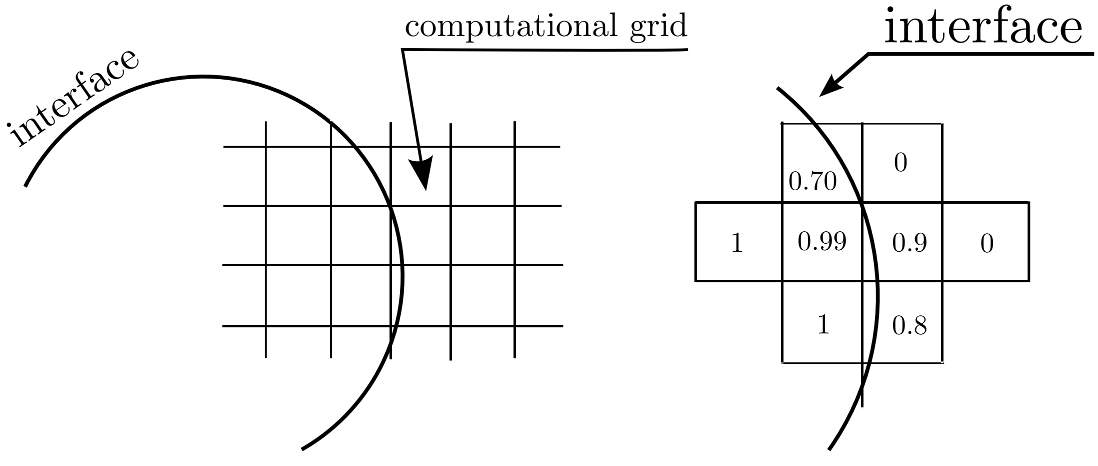
Figure 2.1: Interface representation in two-phase flows. (a) In the *Eulerian* approach, the interface between the fluids is located somewhere in between the computational elements. On the other hand (b) the *Lagrangian* description represents the interface by computational objects, such as nodes, segments and elements, thus achieving a sharp interface.

part of the domain mesh. As can be seen, the interface does not divide any element and no additional treatment is required to deal with high ratio property changes across the interface. Moreover, the interface has zero thickness since it is sharply defined by nodes and triangle edges. The interface is then advected at each time step according to the flow field velocity, thus not requiring any additional equation to describe its motion.

Within the *Eulerian* and *Lagrangian* approaches, different techniques are available to model two-phase flow problems. Each particular method guarantees a set of desired features with different levels of programming efforts. Below, we present an overview of the development of these methods that focus on the *Lagrangian* description, which is part of the present research.

In the 1980's, the *Volume of Fluid*, or VOF, method was proposed by ([40]), which consists of the description of each phase by the volume ratio occupied in the computational cells. The integers 0 and 1 are assigned to define the region occupied by only one of the fluids and values between these integers indicate the region occupied by the interface. Due to the discontinuity of the VOF function, good accuracy cannot be assured when computing the curvature and normal vectors and thus important phenomena are more difficult to capture close to the interface. Figure (2.2) shows a 2-dimensional schematic representation of the VOF method in a structured rectangular grid. The VOF function is assembled by considering the volume occupied by each phase in the computational element.

A 2-dimensional VOF method was developed in [92] and later extended to 3-dimensional flows [91]. The 3-dimensional code included a new methodology based on a two way particle tracking method, which took into account the effects of each phase on the other. Tests were performed to investigate a bubble rising in a stagnant fluid. A wide range of *Eötvös* and *Morton*



(a)

(b)

Figure 2.2: Interface representation in the VOF method. (a) The interface is defined implicitly and it is located somewhere in the computational grid. (b) The VOF function is assembled by considering the volume occupied by each phase at each computational element.

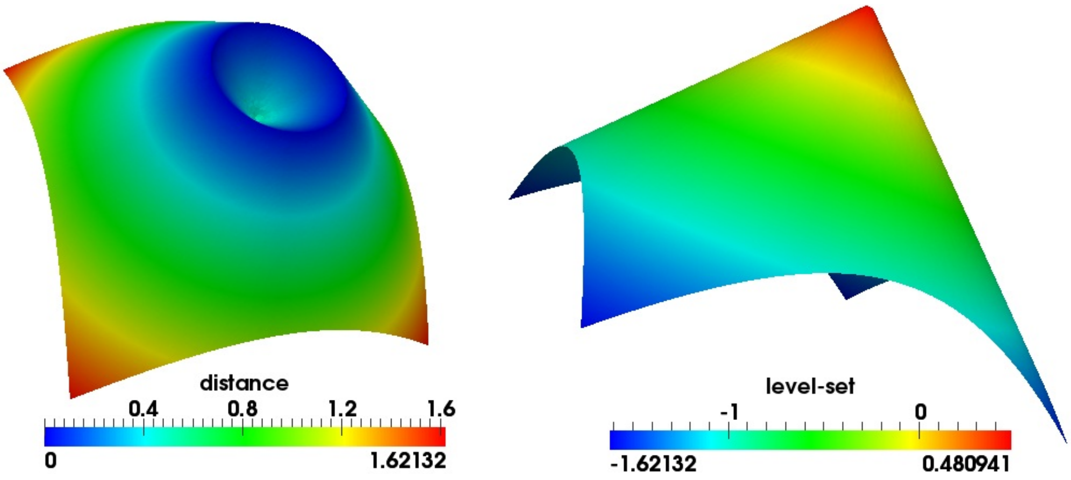
numbers was tested and compared to experimental results, therefore proving that the code was able to capture the effects of different fluid properties.

In ([17]), a two-phase flow model, based on the VOF technique, was developed to simulate flows with high density ratios. They used the continuum surface force model (CSF) to account for the effects of surface tension in the Navier-Stokes equation. Bubble coalescence and bubble test cases were performed to validate their code with available literature data. The proposed model showed good agreement with experimental data.

A 2-dimensional two-phase flow model was implemented in [33] using VOF and a staggered Finite Volume method. This methodology was based on one class of elements commonly found in Finite Element methods, namely Crouzeix-Raviart. Additionally, adaptive mesh refinement was used to obtain uniformity in the mesh elements at the domain of dependence of the interface. They performed static and rising bubble tests and found good agreement to experimental results.

In [99], they developed a 2-dimensional and axisymmetric least-square finite element method to simulate two-phase flows using the VOF method. A modified continuum surface tension model was proposed to treat the surface tension term in the Navier-Stokes equation by calculating the divergence of a stress tensor defined by the gradient of a Heaviside function. The modified model bypasses the difficulty in approximating the curvature of the interface and thus it seems to be easier calculated. Results were successfully compared to different test cases such as a dam breaking, oscillating and stationary bubbles and a conical liquid sheet in a pressure swirl atomizer.

The Level-Set (LS) method in fluid dynamics, first presented by [70], has become an important tool for two-phase flow modeling. The interface is represented by a zero-level distance function that is advected by the flow field. The curvature and the normal vector are conveniently calculated with the aid of the same distance function, and thus a straightforward modeling of surface tension can be achieved. Despite its recent introduction to two-phase flows, the LS method has shown to be one of the most important schemes to model interfacial dynamics. Figure (2.3) shows the representation of the distance function and the level-set function for a 2-dimensional square domain. The distance function is calculated by measuring the distance from each mesh node to its closest interface node, mathematically meaning $\mathbf{x} - \mathbf{x}_I$, where $\mathbf{x}$ is the mesh node and $\mathbf{x}_I$, the interface node. The same distance function is used to compute the level-set function, which is nothing but a signed version of the distance between the mesh nodes and their closest interface nodes. In both cases, the location of the interface is found when the function values match zero.



(a) (b)

Figure 2.3: Representation of the distance and level-set functions. The interface between the fluids is located in the middle of the domain. (a) The distance function is created by computing the distance of each mesh node $\mathbf{x}$ to its closest interface node $\mathbf{x}_I$. (b) The level-set functions is a signed distance function with positive values in one phase and negative values in the other phase.

The LS method in two-phase flows was first developed by [87] and followed by many authors. They proposed a LS scheme with second-order upwind/projection method to compute the solution of the Navier-Stokes equations for immiscible fluids. They shown that the conservation of mass is only guaranteed if the level-set function is re-initialized preferably at every time step. They performed bubble and droplet test cases to validate their new methodology. Additionally, [86] presented an extension of the LS method for 3-dimensional free surface flows. Tests were performed to validate their method including important benchmarks, such as a static droplet and an oscillating drop in zero-gravity, both performed also in this thesis.

The coupling of Level-Set to Finite Elements was presented by [77] in which the fluid flow equation was discretized in a triangular mesh. They proposed a fractional-step method which was used to stabilize the pressure and velocity, therefore allowing the use of the same element for both quantities. Large density ratios were successfully tested and the method was compared to standard benchmark cases.

A stabilized finite element formulation was employed with the level-set method to compute incompressible bubble dynamics in [59]. The streamline upwind Petrov-Galerkin method was used to discretize the Navier-Stokes equation and the continuum surface force model was applied to consider the effects of the surface tension forces in the conservative equations. Bubble coalescence and the rising of a single bubble were studied to test their methodology.

A numerical method for the simulation of three-dimensional incompressible two-phase flows was presented in [56]. The level-set method was used in conjunction with an implicit pressure stabilized finite element method to solve the Navier-Stokes equation. The results were compared to many numerical examples, such as two-phase Poiseuille, Rayleigh-Taylor instability, dam break and sloshing, all showing that the proposed method was accurate to represent the dynamics of two-phase flows.

As is usual with methods based on the Eulerian formulation, the discretization of the Level-Set functions may lead to excessive numerical diffusion, thus requiring its frequent reinitialization and complex advection schemes to prevent such a problem. For example, to overcome the drawbacks found in each methodology, a hybrid method is proposed by [64] and [63] where the curvature and the normal vectors are computed by the Level-Set functions while the interface is captured by the VOF function. Consequently, the mass conservation errors found in the Level-Set formulation and the poor calculation of curvature by the VOF function are avoided.

Unlike VOF and Level-Set methods, the *Lagrangian* description defines the interface between phases explicitly by computational elements. Such a formulation is called *interface tracking*. Regarding the Lagrangian method, *volume-tracking* and *front-tracking* are the most widely used. The former uses marker particles for the reconstruction of the interface, combining precision and accuracy with a relatively low implementation investment ([39],[3]). The latter, first implemented by Glimm and co-authors [34], represent the interface with a set of interconnected nodes which move according to the fluid flow calculated in an *Eulerian* way. Such a description provides a sharp representation of the interface with high accuracy, but its drawback is the need of an explicit treatment of topological changes in the interface, such as for the case of coalescence and break-up of bubbles or drops.

In [95], a new method to simulate unsteady multi-fluid flows was achieved. The method was developed using an Finite Difference approximation in a stationary grid and the interface between the fluids was explicitly represented by a set of geometrical objects such as triangles, segments and points. While the background mesh was fixed in the space, the interface was advected with respect to the flow field. The interface normal vector was found by trigonometric functions in each interface element and the curvature was calculated by an approximation of the Dupin indicatrix of each set of surface nodes. Such an approach leads to a sharp representation of the front whilst, however, the fluid properties are not sharply defined. A smooth function was required close to the interface to avoid numerical instabilities in the transition zone. Even so, the proposed methodology was shown to be suitable to describe bubble dynamics in multiphase flows.

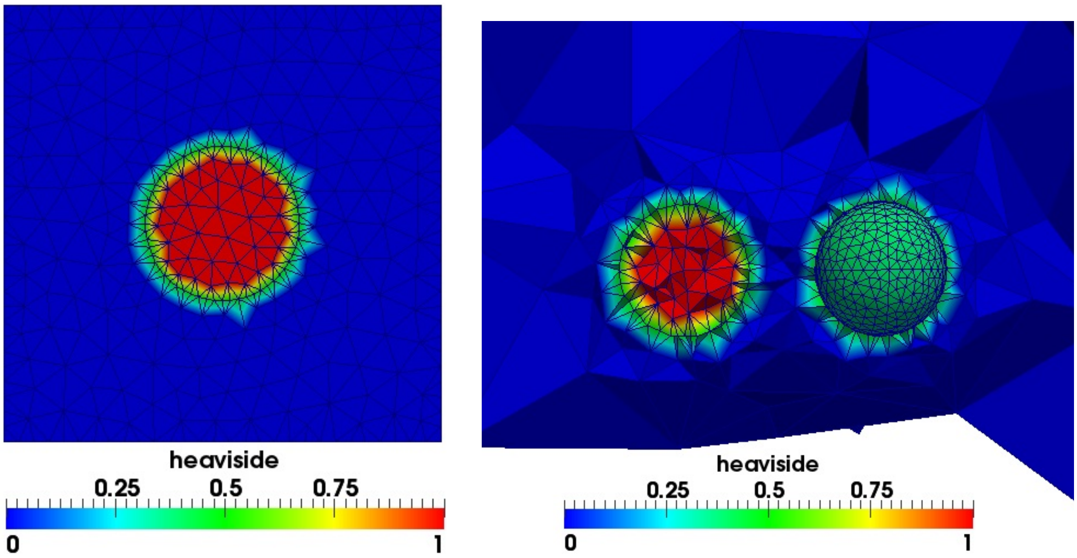
A Lagrangian Level-Set approach was proposed by ([84]) to simulate incompressible two-phase flows. The Navier-Stokes equation was discretized by the Galerkin Finite Element method and the projection method was employed to uncouple the system of non-linear equations. The interface between the fluids was part of the domain mesh and it was discretized by a set of segments and points in 2-dimensional spaces. To reduce mass conservation errors, inherent of the Level-Set method, the methodology employed a Lagrangian technique which moved the nodes of the computational mesh according to the flow field. Moreover, the need of an additional equation to move the front was no longer required. The mesh quality was controlled by common mesh operations such as flipping, insertion and deletion of nodes. In this method the interface was represented by a sub-set of computational nodes and defined by the zero level-set of a function ϕ. Therefore, the curvature and the normal vector were easily calculated by the expression: $\kappa = \nabla \cdot (\nabla\phi / |\nabla\phi|)$ and $\mathbf{n} = \nabla\phi / |\nabla\phi|$, consecutively. Thus, to satisfy the discrete force balance between pressure and surface tension, the gradient of a Heaviside function ∇H was used in substitution of the Dirac Delta function δ found in the Continuum Surface Force (CSF) model. Such an approach is commonly found in *Eulerian* based models, but it has shown to be suitable for interface tracking as well, since the pressure jump condition across the interface was accurately predicted.

The group at the University of Massachusetts has suggested a complete moving mesh technique to simulate free-surface and two-phase flows ([72],[76]) which differs from the fixed background mesh approach. In this case the discretization of the equations was made over an unstructured mesh by an exact fractional step method. Such a technique improves the definition of the interface since the jumps in the fluid properties are kept sharp. Code validations were performed, showing that the proposed methodology could model multiphase flows with accuracy.

The distribution of properties in front-capturing and front-tracking methods may by achieved by a standard step function that changes its value to represent each phase. The Heaviside function $H : \mathbb{R}^n \rightarrow \{0.1\}$ is a commonly used distribution function which is defined within the “one-fluid” context is 0 in one phase and 1 in the other phase. If computational nodes are used to represent the interface between fluids, an average value of 0.5 is commonly assigned to those nodes. Such a function is extremely useful for numerical purposes. It may be used as an auxiliary function to identify in which phase one specific node belongs to, it may also be used to calculate the volume occupied by each phase and to determine whether a node should be inserted or removed. Moreover, the distribution of fluid properties ϕ may be easily computed as:

$$\phi = \phi_1 H + \phi_0 (1 - H) \tag{2.1}$$

where ϕ_0 and ϕ_1 are the fluid properties such as viscosity and density, and H is the Heaviside function. Figure (2.4) shows a distribution of the Heaviside function such that 1 stands for the vapor phase and 0 represents the volume occupied by the liquid phase. In this scheme, the vapor is inside the bubble and the liquid is surrounding the vapor phase.



(a) (b)

Figure 2.4: The Heaviside function is defined as 1 in one fluid and 0 in the other. In the ALE-FE method, an average value of 0.5 is assigned to the nodes belonging the interface. (a) 2-dimensional drop immersed in another fluid. (b) 3-dimensional drops immersed in another fluid. The left drop is sliced to show the nodes belonging the phase ϕ_1.

Due to numerical restrictions found in Level-Set and front-tracking methods, a smoothed version of the Heaviside function may be required. Some authors have suggested different approaches to smooth the sharp edge in the transition zone and this is achieved by defining the Heaviside H as a function of the distance function ϕ:

$$H_\epsilon(\phi) = \begin{cases} 0 & \text{if} \phi < -\epsilon \\ \nu(\phi/\epsilon) & \text{if} -\epsilon \leq \phi \leq \epsilon \\ 1 & \text{if} \phi > \epsilon \end{cases} \tag{2.2}$$

where ϵ is a mesh tolerance, usually defined as a function of the mesh edge length, and $v(\phi/\epsilon)$ is the smoothing in the transition zone. For example, in [93], v is defined as:

$$v(\phi/\epsilon) = \frac{1}{2} + \frac{1}{32}\left(45\phi - 50\phi^3 + 21\phi^5\right) \tag{2.3}$$

and in [86] they adopted the following expression:

$$v(\phi/\epsilon) = \frac{1}{2}\left\{1 + \phi + \frac{1}{\pi}\sin(\pi\phi)\right\} \tag{2.4}$$

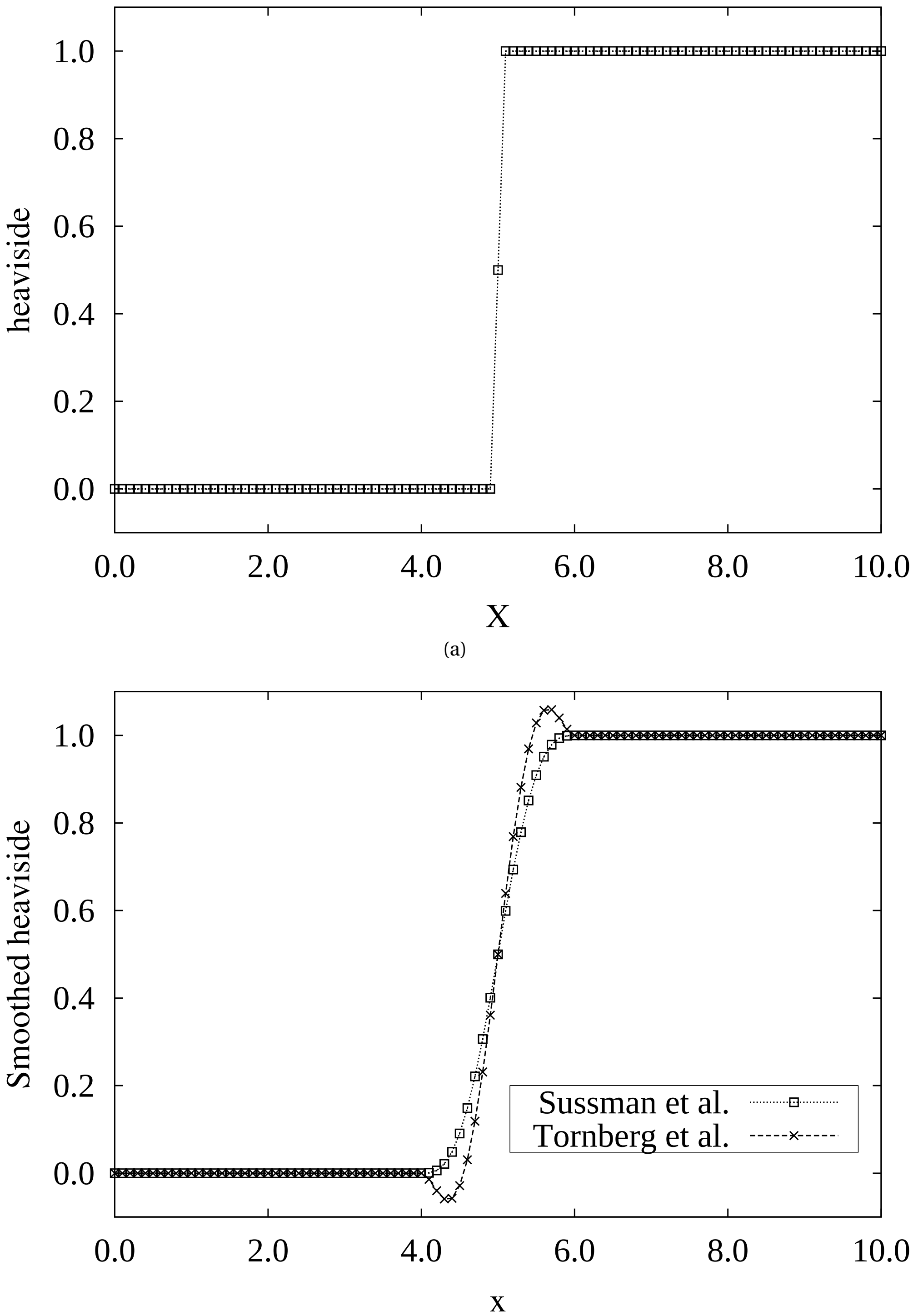
Both smoothing distributions are plotted in Fig. (2.5b). Note that in the present ALE-FE technique, such a smoothing process is not required, thus the method is able to define sharply the distribution of properties in the computational domain. Moreover, another advantage of the definition of the these functions is that the curvature κ and normal vector $\mathbf{n}$ can be easily computed as:

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|} \qquad \text{and} \qquad \kappa = \nabla\cdot\mathbf{n} = \nabla\cdot\frac{\nabla\phi}{|\nabla\phi|} \tag{2.5}$$

where the normal vector n is found by computing the gradient of the Level-Set function ϕ and divided by its module, thus resulting in an unity normal vector. The curvature $k = \nabla\cdot\mathbf{n}$ by definition and, thus can be calculated using the previous defined unit normal vector $\mathbf{n}$. However, as the interface is tracked during the simulation, the function ϕ may not accurately represent the distance from each computational node to the interface. Thus, to keep the precision of the distance function, its initialization is required by imposing $|\nabla\phi| = 1$.

2.3 Adaptive Mesh Refinement

In two-phase flows and in the moving mesh context, the computational nodes are in constant shear due to the moving interface. From one point of view, the ability to translate the mesh nodes from one zone to another brings into the code important features such as low numerical diffusion and sharp representation of the interface. On the other hand, the nodes in the computational domain are often stretched and compressed against each others, and consequently they tend to collapse. Such an inconvenience may shut the simulation down before generating the expected results, an therefore an effective strategy of adaptive mesh refinement should be used to avoid the its interruption. Moreover, the same adaptive strategy may be used to refine or to enlarge particular mesh areas according to the specified local precision required.



(a)

(b)

Figure 2.5: Examples of Heaviside functions applied to Level-Set and Front-Tracking codes. (a) Heaviside with no artificial numerical treatment. (b) Smoothed Heaviside function used in two known references ([86] and [93]).

Typically, the computation begins with a trial solution with a coarse mesh, then the errors of this solution are evaluated and, depending upon the results, new nodes are included or removed from the computational domain.

In the Finite Element method, such adaptive mesh refinement technique may be, basically, divided in two classes such as:

- h - refinement
- p - refinement

In the h-refinement, the interpolation order of the finite elements does not change during the simulation, but instead, their sizes are modified with respect to the simulation requirements. Therefore, high accuracy may be achieved with fewer computational resources. Within the h-refinement class, different techniques are proposed to keep the mesh elements bounded to within a satisfactory aspect ratio. For instance, in the complete re-meshing procedure, the mesh elements are substituted by new elements with different sizes. Such an approach may lead to excessive numerical diffusion, since interpolation is strictly required. However, the higher is the interpolation order used, the lower is the numerical diffusion. Within the h-refinement class, the sub-class r-refinement consists in modify the element connectivity without changing the number of mesh nodes. No interpolation is required, and thus numerical diffusion is avoided. Both procedures may be applied systematically in the computational mesh to achieve reasonable results; however the insertion of nodes are still required to enhance the final mesh.

On the other hand, the p-refinement changes the interpolation order of the finite element to achieve the expected accuracy. Such a procedure may be applied to the entire computational mesh or locally according to the simulation requirements. Such a refinement may be performed hierarchically, thus increasing considerably the accuracy of the simulation. This class of refinements brings into the development of the code another level of complexity, since different interpolation orders are used in the same domain.

Additionally, a combined refinement class may be derived from the above related classes, namely hp-refinement. In such a procedure, the interpolation order of the finite element as well as the element sizes may be changed to achieve the desired precision. It is important to note that all the mentioned classes can be successfully applied in the moving mesh context. In fact, these refinements are extremely useful to keep the computational elements within in a satisfactory shape, consequently allowing the simulation to reach its final stage. These refinement techniques are commonly found in the Finite Element domain, thus highlighting its importance as a numerical tool to model fluid flow problems.

In [10], an error estimation study was performed for h-refinement procedures in 2-dimensional and 3-dimensional meshes, focusing in rectangular and parallelepiped elements. The refinement process began with a coarse mesh and converged to an optimal node distribution according to the proposed geometry. Therefore, the error was evaluated based on the solution convergence.

In [2] proposed an efficient strategy for *hp*-refinement in the Finite Element method where the flexibility of choosing pure *h* or pure *p* is guaranteed by their methodology. Additionally, they suggested two efficient ways to solve large ill-conditioned linear systems of equations which arise from these refinement algorithms.

In [24], an adaptive *hp*-refinement procedure was described for the parallel solution of hyperbolic equations in the rectangular domain. The strategy consisted in using a local Finite Element procedure to preserver high-order accuracy in specific mesh zones. Additionally, an efficient multi-processor data structure was used, which introduced the capability to the code to easily insert and remove elements from the computational mesh.

In [1] a new high-order scheme was proposed, namely k-refinement, in which a faster convergence of the optimal mesh distribution was obtained. They performed many calculations with isogeometric analysis in linear structural and fluid problems to investigate different refinement strategies and compared them to the new proposed *k*-refinement scheme.

In [35], they presented some aspects of a 3-dimensional triangular surface mesh refinement algorithm using object-oriented language. They presented a new implementation of a local refinement algorithm based on 8-subtetrahedron subdivision. The edge division was done in such a way that the mesh quality was kept constant along the whole process, thus achieving an efficient remeshing strategy with low memory usage.

Diagonal flips in triangular surfaces are well discussed in [62]. It has been proven that any two triangulations of a closed surface can be transformed into each other by flipping diagonals in quadrilaterals. An extensive mathematical formulation was presented as the basis to this important surface remeshing procedure such as diagonal flipping and edge contraction, both used in this work.

In [19] and [18], a fast algorithm was proposed to maintain a deformable surface mesh by performing flipping operations, insertion and deletion of points. In plannar triangulations, their algorithm performed edge flipping such that the resulting elements fulfill the requirements for a *Delaunay* triangulation, Moreover, the calculations were performed in linear time, thus substantially decreasing the computational requirements.

2.4 Mass Transfer Models

Two-phase flow problems become even more interesting if phase change occurs, where the mass transfer from one phase to another adds significantly complexity to the dynamics of bubbles and droplets. However, the modeling of such phenomena is not an easy task and should be treated with extreme care. Unfortunately, the related literature, in the numerical domain, is not so widely developed for mass transfer in single and two-phase flows, compared to that for no phase change taking place. Despite the problems related on the modeling of phase change, efforts have been invested to develop tools capable to predict, to a certain extent, boiling and condensation processes in two-phase flows.

The pool boiling process was extensively reviewed by [25], in which 4 basic mechanisms were identified that contribute to the total heat flux, namely evaporation at the liquid interface, enhanced natural convection, natural convection and transient conduction at a nucleation site. It was also found that these mechanisms are strongly linked to the temperature of the superheated wall.

A study of the lateral merging of a vapor bubbles in nucleate pool boiling was performed by [58]. They discretized the Navier-Stokes equation in 3-dimensional space using the Level-Set technique to capture the interface between the phases and the SIMPLE method to solve the equation itself. Systems of single and multiple bubbles were investigated, focusing on how the overall wall heat transfer changes in the merging process. They concluded that the increasing of the wall heat transfer is due to the trapping of a liquid layer between the bubble bases during merger and by drawing of cooler liquid towards the wall during contraction after merger. Tests were performed and correlated well with experimental data.

A numerical modeling of annular film condensation was studied by [61] and [60]. The model was based on a finite volume formulation of the Navier-Stokes and energy equations, in which the interface mass transfer and near-to-wall effects, such as disjoining pressure, were taken into account. They modeled different cross-sectional channel shapes to understand the basic mechanisms involved in the condensation process that takes place at the thin liquid film in two-phase flows. They carried out many code validations with the available literature and obtaining good agreement with experimental data.

In [83], they developed a numerical method for bubble motion with phase change, based on the Level-Set method. In their simulations, they assumed that the temperature in the vapor phase was fixed to be equal the saturation temperature, however such an approach may hide important mechanisms that take place dynamically within the vapor phase. They tested their code with the exact solution of a spherically growing bubble in a superheated liquid, in which a solution is given in cylindrical coordinates.

Boiling flow simulations were performed by [50] within the *OPENFOAM* software. The model was developed with the VOF technique and focused on the investigation of the nucleate boiling process. They included a contact line evaporation model based on the micro region model previous suggested by the same research group. They studied the growth and detachment of a single vapor bubble from a heated steel foil, which seemed to be well modeled. However, comparisons with experimental work were not done.

In [47], they presented a new model to simulate two-phase flows with phase change in two-dimensional domains. The new formulation was included in the previous front-tracking adiabatic code developed by [95] to extend its computation to boiling flows. Since the interface is represented by geometrical objects, special treatment of phase change was considered. They also studied the influence of several parameters in the interface temperature. To validate their model, they compared it to the exact solution of a 1-dimensional test case, followed by the simulation of film boiling with different fluid properties.

Explosive boiling in microgravity was investigated numerically by [29]. They presented a simplified version of the two-phase boiling code developed by [47], which was extended to 3-dimensional spaces. One new aspect compared to the previous developed code is that the saturation temperature was set constant to the interface between fluids. Such an approach leads equally to good results, while however its implementation is easier and the overall calculations are done in a simpler way. They carried out validations and numerical benchmarks to test their code, and finally they studied explosive boiling of a vapor bubble.

2.5 Conclusions

As reported in this chapter, the Finite Element method has been intensively used in fluid dynamics over the past 20 years, proving to be an excellent candidate to solve two-phase problems. Moreover, the *Lagrangian* description was chosen to model the interface between the fluids so that it is sharply represented, ensuring a faithful modeling of the mechanisms involved in bubble flows. Therewith, an new improved adaptive mesh refinement will be proposed here to enforce an optimal mesh quality during the simulations. This proposed work is an extension of the above cited references.

3 Governing Equations

This chapter describes the general equations employed to model the fluid flow. These equations are written in the so-called “one-fluid” formulation using the Arbitrary Lagrangian-Eulerian description in which the computational mesh moves with an arbitrary velocity. Additionally, these equations are re-written in non-dimensional form and their descriptions are detailed. At the end of this chapter, brief comments on the generic initial and boundary conditions are made.

3.1 Introduction

In this work, the fluid is considered to be a *continuum*. This assumption establishes that the smallest volume element considered dV is homogeneous, i.e. the dimension of such an element is large compared to the average distance between the fluid molecules. Thus, the fluid flow may be modeled by the universal conservation laws such as:

- conservation of mass
- conservation of momentum
- conservation of energy

These conservation laws are used to set the general equations for fluid flow problems. However, the modeling of two-phase flows requires an additional description to characterize the different phases involved. In the “one-fluid” approach, one set of equations is utilized to describe the entire domain, and thus an additional marker function is used to modify the properties of the phases. At the interface, a jump condition must be taken into account so that the transition zone located at the interface between the fluids can be modeled. In this way, the mass, momentum and energy conservation laws will be developed by considering only one set of equations, where the different phases are represented by a scalar function which defines different properties, such as viscosity and density. Note that in this formulation, the fluid properties are considered to be constant in each phase with a jump condition at the interface.

3.2 Arbitrary Lagrangian-Eulerian description

In the literature, fluid flow can be expressed by two reference frames commonly used in fluid dynamics, namely *Eulerian* and *Lagrangian* descriptions. The former describes the fluid motion by a fixed referential frame where the continuum moves with respect to the mesh nodes. The later describes the fluid flow through the material derivative, i.e. the referential frame is moving according to the fluid motion. A more generalized way to describe the fluid motion may consider the referential frame not fixed space or moving with the same velocity of the fluid motion, but instead the referential frame moves with an arbitrary velocity that does not necessary represent any of the standard description. Such a generalized representation of the flow field is referred as the *Arbitrary Lagrangian-Eulerian* description or simply ALE.

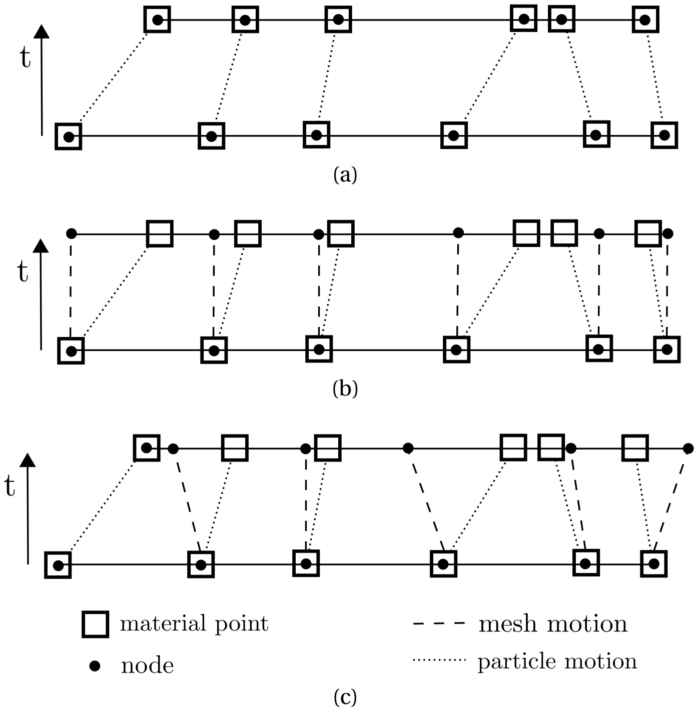
Figure (3.1) depicts a schematic representation of an one-dimensional domain with the three mentioned descriptions. In Fig. (3.1a), the nodes are moved with the same velocity as the flow field, thus the material points are found to be located on the mesh nodes. As can be seen, depending on the flow conditions, in a short period of time the nodes may be poorly distributed, consequently degrading the accuracy of the solution. In Fig. (3.1b), the mesh nodes are fixed in the space domain and the particle motion is evaluated according to their position and velocity and then interpolated back to the mesh nodes. The adverse condition here is that the quantity in the next time step is interpolated, and thus numerical diffusion may spoil the accuracy of the simulation. Figure. (3.1c) shows an example of the ALE motion, where the material points are calculated based on the arbitrary motion of the mesh nodes. Such a inherent capability of the ALE description allows the mesh nodes to be repositioned according to some refinement criteria, thus avoiding the shortcomings of the pure *Eulerian* and *Lagrangian* descriptions.

According to [45], the ALE description can be represented by three different domains as illustrated in Fig. (3.2). These domains are divided as: material domain ($\mathbf{X}$), spatial domain ($\mathbf{x}$) and referential domain ($\tilde{\mathbf{X}}$). Both the material and referential domains are moving while the spatial domains is fixed. The velocity of a particle that travels from the material domain to the spatial domain may be written according to the operator ϕ which describes the motion as:

$$\mathbf{v} = \frac{\partial \mathbf{x}}{\partial t} = \frac{\partial}{\partial t} X(\mathbf{x}, t) \tag{3.1}$$

On the other hand, the velocity of a particle that travels from the referential domain to the spatial domain according to the operator $\tilde{\phi}$ may be written as:

$$\hat{\mathbf{v}} = \frac{\partial \mathbf{x}}{\partial t} = \frac{\partial}{\partial t} \tilde{X}(\mathbf{x}, t) \tag{3.2}$$



(a)

(b)

(c)

Figure 3.1: One-dimensional examples of the (a) *Lagrangian* description, in which the mesh nodes move according to the flow field, (b) *Eulerian*, in which the mesh is fixed in the space and (c) ALE, in which a generalized description is achieved.

Consequently, the two velocities $\mathbf{v}$ and $\tilde{\mathbf{v}}$ are mapped into the spatial domain and thus can be rewritten to described the Arbitrary Lagrangian-Eulerian motion as a subtraction of these velocities. Let us consider f to be a quantity in the space-time domain expressed as a function of these two mentioned descriptions. The resulting scheme for the quantity f is given by:

$$\frac{Df}{Dt} = \frac{\partial f}{\partial t} + ((\mathbf{v} - \hat{\mathbf{v}}) \cdot \nabla) f = \frac{\partial f}{\partial t} + (\mathbf{c} \cdot \nabla) f \tag{3.3}$$

In this scheme, if the mesh velocity is $\tilde{\mathbf{v}} = \mathbf{v}$, the time variation of the quantity f is exactly the same in two different time steps, thus the *Lagrangian* description is recovered. By setting the mesh velocity $\tilde{\mathbf{v}} = 0$ instead, the referential domain is fixed in the space, while the quantity f varies, thus the pure *Eulerian* description is achieved.

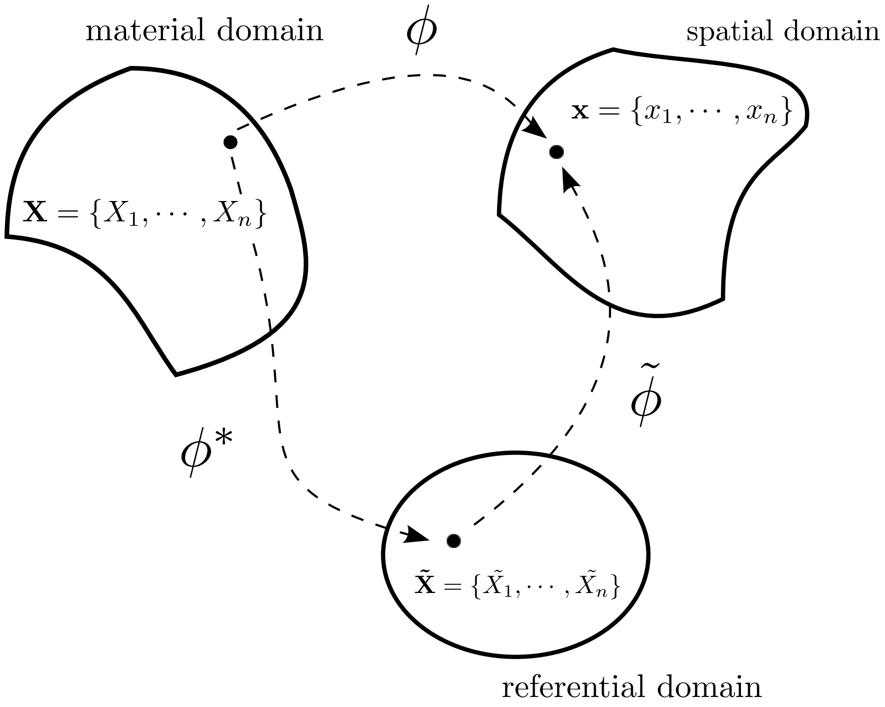

material domain
ϕ
spatial domain
$\mathbf{x} = \{x_1, \cdots, x_n\}$
$\mathbf{X} = \{X_1, \cdots, X_n\}$
$\tilde{\phi}$
ϕ^*
$\tilde{\mathbf{X}} = \{\tilde{X}_1, \cdots, \tilde{X}_n\}$
referential domain

Figure 3.2: Material, referential and spatial configuration for the Arbitrary Lagrangian-Eulerian framework.

3.3 Conservation of mass

The conservation of mass establishes that for a given fluid with specific weight ρ flowing through a fixed volume element V, the mass entering per unit time is equal to the mass departing per unit time plus the increase in mass in the control volume per unit time. Mathematically this can be described as follows:

$$\int_V \frac{\partial}{\partial t} dm \tag{3.4}$$

where the infinitesimal form $dm = \rho dv$ is obtained by substitution into Eq. (3.4), so that:

$$\int_V \frac{\partial}{\partial t} dm = \int_V \frac{\partial}{\partial t}(\rho dV) = \int_V \frac{\partial \rho}{\partial t} dV + \int_V \rho \frac{\partial dV}{\partial t} = \int_V \frac{\partial \rho}{\partial t} dV \tag{3.5}$$

The mass net flux is defined as follows:

$$\oint_S \rho \mathbf{v} \cdot \mathbf{n} dA \tag{3.6}$$

where **v** is the fluid velocity and **n** is the normal component which is parallel to the flow. Thus, the integral form of mass conservation is written as:

$$\int_V \frac{\partial \rho}{\partial t} dV = -\oint_S \rho \mathbf{v} \cdot \mathbf{n} dA \tag{3.7}$$

On the right hand side of the above equation, the surface integral can be changed to a volume integral by the Gauss theorem (see appendix A), thus Eq. (3.7) becomes:

$$\int_V \frac{\partial \rho}{\partial t} dV = -\int_V \nabla \cdot (\rho \mathbf{v})\ dV \tag{3.8}$$

or:

$$\int_V \left[\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) \right] dV = 0 \tag{3.9}$$

this equation is valid for any volume element V. Thus, for an infinitesimal volume dV, it is:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{3.10}$$

In the present work, the conservation of mass is described in 3-dimensional cartesian coordinates so that:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho v_x}{\partial x} + \frac{\partial \rho v_y}{\partial y} + \frac{\partial \rho v_z}{\partial z} = 0 \tag{3.11}$$

Moreover, the incompressibility hypotheses may be applied to the term $\partial\rho/\partial t$, since the fluid density remains constant in each phase. The conservation of mass is thus reduced to:

$$\rho\Big[\frac{\partial\rho v_x}{\partial x}+\frac{\partial\rho v_y}{\partial y}+\frac{\partial\rho v_z}{\partial z}\Big]=\rho\nabla\cdot\mathbf{v}=0 \tag{3.12}$$

Since $\rho \neq 0$ for all phases, the final equation becomes:

$$\nabla\cdot\mathbf{v}=0 \tag{3.13}$$

Equation (3.13) represents the incompressibility condition of a fluid. If this fluid is ruled by such a condition, it is said to be incompressible. This equation is also called the *continuity equation.*

3.4 Conservation of momentum

The same concept of conservation of mass is applied to the conservation of momentum to achieve a mathematical formulation. For a given fluid with density ρ flowing through a volume element dV, the mass entering per unit time is equal to the mass departing per unit time plus the increase in mass in the volume element per unit time, mathematically this can be defined as:

$$\int_V \frac{\partial(\rho\mathbf{v})}{\partial t}\,dV \tag{3.14}$$

From Eq. (3.6), the mass flux through the volume element dV is known, thus multiplying the integral term by a velocity vector in the x direction, the momentum flux crossing the element's area is given as:

$$\oint_S \rho\mathbf{v}\mathbf{v}\cdot\mathbf{n}\,dA \tag{3.15}$$

The resulting surface forces are given by:

$$\oint_S \sigma \cdot \mathbf{n}\, dA \tag{3.16}$$

where σ is the stress tensor acting at each area element. Finally, the volume forces, which are represented by the gravity and surface tension forces, are:

$$\int_V \rho \mathbf{g}\, dV \qquad \text{and} \qquad \int_V \mathbf{f}\, dV \tag{3.17}$$

Using the Continuum Surface Force (CSF) model according to [12], the source term **f** in Eq. (3.17) can be written as:

$$\mathbf{f} = \sigma \kappa \delta \mathbf{n} \tag{3.18}$$

where κ is the curvature , σ is the surface tension and **n** is the surface unity outward normal vector. Additionally, δ represents the Dirac delta function with support on the interface. Thus, the surface force **f** may be written as:

$$\int_V \mathbf{f}\, dV = \int_V \sigma \kappa \mathbf{n} \delta\ dV \tag{3.19}$$

The surface tension force in Eq. (3.19) will be treated as **f** in the following equations to simplify the notation. However, it will be detailed later, in chapter 5. Therefore, the conservation of momentum in the Arbitrary Eulerian-Lagrangian description is therefore described as follows:

$$\int_V \frac{\partial(\rho \mathbf{v})}{\partial t}\, dV = -\oint_S \rho \mathbf{c}(\mathbf{v} \cdot \mathbf{n})\, dA + \oint_S \sigma \cdot \mathbf{n}\, dA + \int_V \rho\ \mathbf{g} dV + \int_V \mathbf{f} dV \tag{3.20}$$

The Gauss theorem is applied to the surface integrals to obtain the corresponding volume integrals, and thus the integral form of the conservation of momentum is:

$$\int_V \frac{\partial(\rho \mathbf{v})}{\partial t}\, dV = -\int_V \nabla \cdot (\rho \mathbf{c} \mathbf{v})\, dV + \int_V \nabla \cdot (\sigma)\, dV + \int_V \rho \mathbf{g}\, dV + \int_V \mathbf{f}\, dV \tag{3.21}$$

Similar to the equation for conservation of mass, Eq. (3.21) is valid for any volume element V. Thus, for an infinitesimal volume dV, it can be expressed as:

$$\frac{\partial(\rho\mathbf{v})}{\partial t}+\nabla\cdot(\rho\mathbf{c}\mathbf{v})=\nabla\cdot(\sigma)+\rho\mathbf{g}+\mathbf{f} \tag{3.22}$$

and expanding the left hand side terms, this becomes:

$$\rho\frac{\partial\mathbf{v}}{\partial t}+\mathbf{v}\frac{\partial\rho}{\partial t}+\rho\mathbf{c}\cdot\nabla\mathbf{v}+\mathbf{c}\cdot\nabla(\rho\mathbf{v})=\rho\left[\frac{\partial\mathbf{v}}{\partial t}+\mathbf{c}\cdot\nabla\right]+\mathbf{c}\left[\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\mathbf{v})\right] \tag{3.23}$$

The term $\partial\rho/\partial t+\nabla\cdot(\rho\mathbf{v})$ is null due to the continuity equation. Thus, rearranging the terms, the momentum equation is written as:

$$\rho\left[\frac{\partial\mathbf{v}}{\partial t}+\mathbf{c}\cdot\nabla\mathbf{v}\right]=\nabla\cdot\sigma+\rho\mathbf{g}+\mathbf{f} \tag{3.24}$$

The total stress tensor σ in the above formulation is defined as:

$$\sigma=-p\mathbf{I}+\tau \tag{3.25}$$

where p stands for the pressure, $\mathbf{I}$ the identity matrix and τ the deviatoric stress tensor. Substituting the right hand side of Eq. (3.25) into the momentum equation, the (Eq. 3.24) becomes:

$$\rho\left[\frac{\partial\mathbf{v}}{\partial t}+\mathbf{c}\cdot\nabla\mathbf{v}\right]=-\nabla p+\nabla\cdot\tau+\rho\mathbf{g}+\mathbf{f} \tag{3.26}$$

The deviatoric stress tensor term τ represents the deformation of a fluid element in the presence of a velocity gradient. The constitutive relation of such a tensor changes according to the physical properties of the fluid. All the fluids considered in the present work have a shear stress proportional to their angular deformation, i.e. the fluids are said to be *Newtonian*. Thus,

τ may be written as:

$$\tau = 2\mu\mathbf{D} + \left(\lambda - \frac{2}{3}\mu\right)(\nabla \cdot \mathbf{v})\mathbf{I} \tag{3.27}$$

where μ is the dynamic viscosity and λ the volumetric viscosity of the related fluid. Due to the incompressibility condition of Eq. (3.13), the deviatoric stress tensor τ may be rewritten as:

$$\tau = 2\mu\mathbf{D} \tag{3.28}$$

The strain tensor **D** in the above equation is given by:

$$\mathbf{D} = \frac{1}{2}\dot{\gamma}(\mathbf{v}) = \frac{1}{2}(\nabla\mathbf{v} + \nabla\mathbf{v}^T) \tag{3.29}$$

where $\dot{\gamma}(\mathbf{v})$ it the rate-of-strain tensor. Thus, the divergence of σ may be calculated as follows:

$$\nabla \cdot \sigma = \nabla \cdot [-p\mathbf{I} + \mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)] = -\nabla p + \nabla \cdot [\mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)] \tag{3.30}$$

The momentum equation in the ALE description can now be written as:

$$\rho\left[\frac{\partial\mathbf{v}}{\partial t} + \mathbf{c} \cdot \nabla\mathbf{v}\right] = -\nabla p + \nabla \cdot [\mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)] + \rho\mathbf{g} + \mathbf{f} \tag{3.31}$$

Rewriting the above equation using the *substantial derivative*, this yields:

$$\rho\frac{D\mathbf{v}}{Dt} = -\nabla p + \nabla \cdot [\mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T)] + \rho\mathbf{g} + \mathbf{f} \tag{3.32}$$

3.5 Conservation of energy

The energy (heat) transport states that for a given fluid with density ρ, flowing through a volume element V, the accumulating mass rate of energy going into the volume element per time unit is equal to the net mass flux leaving the volume element, in the absence of a source term. Mathematically, the accumulating mass rate of energy is described as follows:

$$\int_V \frac{\partial T}{\partial t} dV \tag{3.33}$$

where T may represent any scalar per volume unit. In the present context, T stands for the temperature distribution of the fluid. Thus, the net flux due to the heat transport going into and out of an element volume V is defined as:

$$\oint_S T\mathbf{v}\cdot\mathbf{n}\,dA \tag{3.34}$$

where T is the temperature of the fluid, $\mathbf{v}$ is the fluid velocity and n is the normal component which is parallel to the flow field. The net heat flux leaving the volume element due to conduction is given by:

$$\oint_S \mathbf{J}\cdot\mathbf{n}\,dA \tag{3.35}$$

Thus, the integral form of the heat transport is derived as:

$$\int_V \frac{\partial(\rho c_p T)}{\partial t} dV = -\oint_S (\rho c_p T)\mathbf{v}\cdot\mathbf{n}\,dA - \oint_S \mathbf{J}\cdot\mathbf{n}\,dA \tag{3.36}$$

where ρ is the fluid density and c_p is the specific heat. The Gauss Theorem is again applied to the surface integrals in the above equation to obtain the corresponding volume integrals, so that:

$$\int_V \frac{\partial(\rho c_p T)}{\partial t} dV = -\int_V \nabla \cdot (\rho c_p T)\,\mathbf{v}\,dV - \int_V \mathbf{J}\sigma\,dV \tag{3.37}$$

or:

$$\int_V \rho c_p \left[\frac{\partial T}{\partial t} + \nabla \cdot (T\mathbf{v})\right] dV = -\int_V \nabla \cdot (\mathbf{J})\ dV \tag{3.38}$$

This equation is valid to any volume element V. Thus, for a infinitesimal volume dV, it can be written as:

$$\rho c_p \left[\frac{\partial T}{\partial t} + \nabla \cdot (T\mathbf{v})\right] = -\nabla \cdot \mathbf{J} \tag{3.39}$$

Expanding the left hand side terms of Eq. (3.39), this gives:

$$\rho c_p \left[\frac{\partial T}{\partial t} + T\nabla \cdot \mathbf{v} + \mathbf{c} \cdot \nabla \cdot T\right] = -\nabla \cdot \mathbf{J} \tag{3.40}$$

The term $T\nabla \cdot \mathbf{v}$ is null due to the incompressibility condition given by the continuity equation (Eq. 3.13) and $\mathbf{c}$ is the relative velocity in the ALE description. Considering that the diffusive flux is given by Fourier's law of heat conduction, thus:

$$\mathbf{J} = -k\nabla T \tag{3.41}$$

where k is the material's thermal conductivity. Thus, the Eq. (3.40) results in:

$$\rho c_p \left[\frac{\partial T}{\partial t} + \mathbf{c} \cdot \nabla T\right] = \nabla \cdot (k\nabla T) \tag{3.42}$$

Rewriting the above equation using the *substantial derivative* yields:

$$\rho c_p \frac{DT}{Dt} = \nabla \cdot (k \nabla T) \tag{3.43}$$

This equation describes the heat transport in a fluid. The left hand side term represents the heat convection and the right hand side term stands for the diffusive effect. Note that if k assumes large values, the heat transport by the convective effect becomes less important compared to the diffusive effect. On the other hand, for low values of k, the heat transport by diffusion may be neglected.

3.6 Non-dimensional form of the Navier-Stokes equation

This section describes the non-dimensionalization of the continuity, momentum and energy equations. Such a procedure is not required to solve the equations, however a better understanding of the influence of each term on the simulated flow is easier t oaccess. Two distinct non-dimensional equations will be presented. The first is used when the flow velocity and the characteristic length are known and both are set as simulation parameters. The second is employed for gravity driven flows, where the flow velocity is not a known parameter. Therewith, the gravity should be taken as a referential parameter, so that the velocity is a function of the referential length and the acceleration of gravity.

The continuity and momentum equations are non-dimensionalized by defining the following non-dimensional parameters:

$$\begin{array}{llll} \rho = \rho_\infty \rho^* & x = L x^* & \mathbf{g} = g_\infty \mathbf{g}^* & \mathbf{c} = U \mathbf{c}^* \\ \mu = \mu_\infty \mu^* & \mathbf{v} = U \mathbf{v}^* & t = \dfrac{L}{U} t^* & \kappa = \dfrac{1}{L} \kappa^* \\ p = \rho_\infty U^2 p^* & \dfrac{\partial}{\partial t} = \dfrac{U}{L} \dfrac{\partial}{\partial t^*} & \nabla = \dfrac{1}{L} \nabla^* & \sigma = \sigma_0 \sigma^* \end{array}$$

where the superscript $*$ stands for the non-dimensional variables. By substituting the above listed parameters into the continuity equation (Eq. (3.13)), the non-dimensionalization process yields:

$$\frac{U}{L}\nabla^* \cdot \mathbf{v}^* = 0 \tag{3.44}$$

Multiplying the equation above by U/L, the non-dimensional form of the continuity equation turns simply into:

$$\nabla^* \cdot \mathbf{v}^* = 0 \tag{3.45}$$

The same procedure is applied to the conservation of momentum, substituting the non-dimensional parameters into Eq. (3.31):

$$\begin{aligned}\rho^* \left[\frac{\rho_0 U^2}{L}\frac{\partial \mathbf{v}^*}{\partial t^*} + \frac{\rho_0 U^2}{L}\mathbf{v}^* \cdot \nabla^* \mathbf{v}^*\right] = &-\frac{\rho_0 U^2}{L}\nabla^* p^* \\ &+ \frac{\mu_0 U}{L^2}\nabla^* \cdot [\mu^* (\nabla^* \mathbf{v}^* + \nabla^* \mathbf{v}^{*T})] + \rho_0 g_0 \rho \mathbf{g}^* + \frac{\sigma_0}{L^2}\mathbf{f}\end{aligned} \tag{3.46}$$

Multiplying the above equation by $L/\rho_0 U^2$ yields:

$$\frac{\partial \mathbf{v}^*}{\partial t^*} + \mathbf{v}^* \cdot \nabla^* \mathbf{v}^* = -\frac{1}{\rho^*}\nabla^* p^* + \frac{\mu_0}{\rho_0 L U}\nabla^* \cdot [\mu^* (\nabla^* \mathbf{v}^* + \nabla^* \mathbf{v}^{*T})] + \frac{g_0 L}{U^2}\mathbf{g}^* \frac{\sigma_0}{\rho_0 L U^2}\mathbf{f} \tag{3.47}$$

which is the non-dimensional form of the momentum equation. Three important non-dimensional groups are found in this equation, namely the *Reynolds*, the *Froude* and the *We* numbers. Their descriptions are given below:

- **Reynolds number** (Re): ratio of the inertia force to the viscous force, given by:

$$Re = \frac{\rho_0 L U}{\mu_0} = \frac{L U}{\nu_0}\ , \tag{3.48}$$

where ρ_0, L, U and μ_0 are reference values for density, length, velocity and viscosity, respectively. ν_0 stands for the kinematic viscosity which is the ratio between the dynamic viscosity μ_0 and density ρ_0. If $Re < 1$, the viscous forces are stronger than the inertial ones. In the other hand, when $Re \gg 1$, the inertial forces are predominant. Moreover, the *Reynolds* number is useful to characterize the flow, which can be laminar if ($Re < Re_{\text{critical}}$) or turbulent if ($Re > Re_{\text{critical}}$).

- **Froude number** (Fr): ratio of the inertia force to the gravitational force, given by:

$$Fr = \frac{U}{\sqrt{gL}} \tag{3.49}$$

  where g is the referential gravity. This number is useful to characterize flows in which the gravitational effects can not be neglected, for instance when an air bubble is immersed in another fluid with different densities. If $Fr < 1$, the gravity effect has more influence than the inertial force.

- **Weber number** (We): ratio of the inertia force to the surface tension force, given by:

$$We = \frac{\rho_0 L U^2}{\sigma_0} \tag{3.50}$$

  where σ_0 is the referential surface tension. This number is useful to characterize flows in which the interfacial forces can not be neglected. However, if $We << 1$, the surface tension does not play an important role in the flow field. Since as will be seen, We appears as $\frac{1}{We}$ in the momentum equation.

The non-dimensional equations presented above consider that the flow velocity is a known parameter, thus it is taken as a system referential. However, in certain cases where the velocity is a consequence of a condition and it is not imposed directly on the system, a different referential parameter should be used instead. For instance, consider the rising of a single bubble immersed in another fluid driven only by buoyancy effect. In this case, the velocity field is the result of the system and it can not be used as parameter. Instead, a different system referential is required. Since the gravity plays an important role in this case, it may be used as system referential. Thus, the velocity field $\mathbf{v}$ is commonly non-dimensionalized by $\sqrt{gD}$ and the characteristic length L by the bubble's diameter D. Substituting these parameters into Eq. (3.47), the momentum equation becomes:

$$\begin{aligned} \rho^* \left[\frac{\rho_0 g_0 D}{D} \frac{\partial \mathbf{v}^*}{\partial t^*} + \frac{\rho_0 g_0 D}{D} \mathbf{v}^* \cdot \nabla^* \mathbf{v}^* \right] &= - \frac{\rho_0 g_0 D}{D} \nabla^* p^* \\ &+ \frac{\mu_0 \sqrt{g_0 D}}{D^2} \nabla^* \cdot [\mu^* (\nabla^* \mathbf{v}^* + \nabla^* \mathbf{v}^{*T})] + \rho_0 g_0 \, \rho^* \mathbf{g}^* + \frac{\sigma_0}{D^2} \mathbf{f}^* \end{aligned} \tag{3.51}$$

Multiplying the above equation by $1/\rho_0 g_0$ and dropping all the superscripts $*$, the non-dimensional momentum equation becomes:

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\frac{1}{\rho} \nabla p + \frac{\mu_0 \sqrt{g_0 D}}{D^2} \nabla \cdot [\mu(\nabla \mathbf{v} + \nabla \mathbf{v}^T)] + \rho_0 g_0 \rho g + \frac{\sigma_0}{\rho_0 g_0 D^2} \mathbf{f} \tag{3.52}$$

Two important non-dimensional groups are found in the above equation, namely the *Archimedes* and the *Eötvös* numbers, in which the descriptions are written below:

- **Archimedes number** (N): ratio of the buoyancy force to the viscous force, given by:

  $$N = \frac{\rho_0^2 D^3 g}{\mu_0^2}\ , \tag{3.53}$$

  where ρ_0, D, g and μ_0 are reference values for density, bubble/drop diameter, gravity and viscosity, respectively.

- **Eötvös number** (Eo): ratio of the buoyancy force to the surface tension force, given by:

  $$Eo = \frac{\rho_0 g D^2}{\sigma_0} \tag{3.54}$$

  where $sigma_0$ is the referential surface tension coefficient. This number is useful to measure quantitatively the effects of surface tension and gravity. Is is equivalent to the *We* number but the velocity is non-dimensionalized as $\sqrt{gD}$ instead.

Additionally, the non-dimensional *Morton* number and *Capillary* number are defined due to their importance to characterize the shape of a bubble or (drop) and the relative effect of the viscous and surface tension forces, respectively:

- **Morton number** (M): ratio of viscous, capillary and gravitational forces, given by:

  $$M = \frac{\mu_0^4 g}{\rho_0 \sigma_0^3} = \frac{Eo^3}{N^2} = \frac{We^3}{Fr Re^4} \tag{3.55}$$

  This number may be expressed by the ratio of the *Eötvös* number raised to the third power relative to the *Archimedes* number to the second power.

- **Capillary number** (Ca): ratio of viscous and surface tension forces, given by:

  $$Ca = \frac{\mu U}{\sigma_0} \tag{3.56}$$

  If $Ca << 1$, the flow is dominated by capillary forces. This number also represents the ratio of the *Weber* number We and the *Reynolds* number Re, thus it correlates the inertia, viscous and interfatial forces. Such a number is an important parameter to characterize two-phase flow regimes.

A similar procedure is applied to the energy equation to achieve its non-dimensional form. Thus, beside using the same parameters of the continuity and momentum equations above, a new parameter is defined to non-dimensionalize the temperature:

$$T^* = \frac{T - T_s}{T_w - T_s} \Longrightarrow T = (T_w - T_s)T^* + T_s$$

where T_w is the temperature in the wall and T_s represent the saturation temperature. Thus, substituting the non-dimensional parameters into Eq. (3.43):

$$\frac{\rho c_p U}{L}\frac{\partial}{\partial t^*}[(T_s - T_\infty)T^* + T_\infty] + \rho c_p U \mathbf{c}^* \cdot \frac{1}{L}\nabla^*[(T_s - T_\infty)T^*] = \frac{1}{L}\nabla^* \cdot \left[k_\infty k^* \frac{1}{L}\nabla^*(T_s - T_\infty)T^*\right] \tag{3.57}$$

$$\frac{\rho c_p U}{L}(T_s - T_\infty)\frac{\partial T^*}{\partial t^*} + \frac{\rho c_p U}{L}(T_s - T_\infty)\mathbf{c}^* \cdot \nabla^* T^* = \frac{k_\infty}{L^2}(T_s - T_\infty)\nabla^* \cdot (k^* \nabla T^*) \tag{3.58}$$

Multiplying the above mentioned equation by $L/[(T_w - T_s)U\rho c_p]$, this yields:

$$\frac{\partial T^*}{\partial t^*} + \mathbf{c}^* \cdot \nabla^* T^* = \frac{k_\infty}{\rho c_p U L}\nabla^* \cdot (k^* \nabla^* T^*) \tag{3.59}$$

In Eq. (3.59) the non-dimensional group is defined as:

- **Peclet number** (Pe): ratio of the characteristic body length to the thickness of the thermal boundary layer, given as:

$$Pe = \frac{\rho c_p UL}{k_\infty} \tag{3.60}$$

U stands for the referential velocity, L represents the referential length and k_∞ is the thermal conductivity. This number is used to characterize the effects of thermal diffusion and convection. If $Pe < 1$, the thermal diffusion effect is more pronounced than the convection.

For hydrodynamic problems where the fluid velocity is imposed on the system and consequently Re is defined, it is convenient to split the non-dimensional number Pe such that:

$$Pe = \frac{UL}{k_\infty} = \frac{\rho_0 UL}{\mu_0}\frac{\mu_0}{\rho_0 k_\infty} = RePr \tag{3.61}$$

In the above equation, the non-dimensional group is defined as:

- **Prandtl number** (Pr): ratio of thickness of the hydrodynamic to the temperature boundary layers, given as:

  $$Pr = \frac{\mu_0}{\rho_0 k_\infty} \tag{3.62}$$

  If $Pr < 1$ in a given fluid, the thermal diffusion occurs at a greater rate than the momentum diffusion. Therefore, the effect of heat conduction is stronger compared to heat convection. For gases, this number is on the order of 0.7 to 1.0. In liquids, this number is usually larger than for gases and the higher is the viscosity, the higher is the Pr number.

Thus, substituting the non-dimensional groups Re and Pr into Eq. (3.59) and dropping all the subscripts, the heat transport equation becomes:

$$\frac{\partial T}{\partial t} + \mathbf{c}\cdot\nabla T = \frac{1}{RePr}\nabla\cdot(k\nabla T) \tag{3.63}$$

This equation describes the transient distribution of temperature in a fluid. The aforementioned non-dimensional equations are discretized by the Finite Element method, which will be described in the Chapter (4).

3.6.1 Mass Transfer

In order to include the mass transfer effect to the Navier-Stokes and Heat transport equations, an extensive reformulation is required. Let us consider one fluid occupying the computational domain in which two distinct phases are present, namely the liquid phase and its vapor. As mentioned earlier in this chapter, only one set of equations is used to model the entire domain, consequently these two phases are also modeled by the same set of equations within the "one-fluid" formulation. If boiling or condensation occurs, the volume occupied by each phase changes, however the overall volume is kept constant. In our model, this can be achieved by considering that the field remains incompressible, but a volume change (compressibility condition) is allowed at the interface. This formulation is based on the work of [29] and [47], and thus extended here for the present ALE-FE code. The heat transport equation (Eq. (3.43) is modified and rewritten as follows:

$$\rho c_p \frac{DT}{Dt} = \nabla \cdot (k \nabla T) + Z\dot{q}|\nabla H| \tag{3.64}$$

where Z is a constant which modifies the latent heat h_{fg} as a result of unequal specific heats. $|\nabla H|$ is the module of the gradient of the Heaviside function H, which defines the region for the volume changing. The heat flux $\dot{q}$ is given by:

$$\dot{q} = \nabla^2 T \tag{3.65}$$

where T is the temperature distribution in the computational domain. In [29], the heat flux $\dot{q}$ was found using two normal probes which originated at the phase boundary and extend a certain distance into the vapor and the liquid. However in the Finite Element formulation, the Eq. (3.65) can be used, since the operator ∇^2 is already assembled.

We assume that the temperature of the interface between the phases T_I is equal to the saturation temperature T_s at the system pressure p_{sys} as follows:

$$T_I = T_s(p_{sys}) \tag{3.66}$$

Since the velocity field is not incompressible at the whole computational domain, it is necessary to include the volume change occurring at the interface into Eq. (3.13). Thus, it needs to be slightly modified to accommodate this effect by rewriting the field velocity $\mathbf{v}$ as a function of the vapor velocity $\mathbf{v}_v$, the liquid velocity $\mathbf{v}_l$ and the Heaviside function H. In order to achieve the formulation, we use the same distribution function for fluid properties of Eq. (2.1) as follows:

$$\mathbf{v} = \mathbf{v}_v H + \mathbf{v}_l (1 - H) \tag{3.67}$$

Considering $\nabla \cdot \mathbf{v}_v = \nabla \cdot \mathbf{v}_l = 0$, and the mass transfer rate $\dot{m}_I$ at the phase boundary as follows:

$$\dot{m}_I = \rho_l(\mathbf{v}_l - \mathbf{v}_I) = \rho_v(\mathbf{v}_v - \mathbf{v}_I) = \frac{\dot{q}}{h_{fg}} \tag{3.68}$$

where $\mathbf{v}_I$ is the normal velocity of the interface. The divergence of the velocity field $\mathbf{v}$ is given by:

$$\nabla \mathbf{v} = \frac{1}{h_{fg}} \left(\frac{1}{\rho_v} - \frac{1}{\rho_l} \right) \dot{q} |\nabla H| \tag{3.69}$$

The normal interface velocity $\mathbf{v}_I$ is found to be:

$$\mathbf{v}_I = \mathbf{v} + \dot{m}\beta \tag{3.70}$$

In [29], the value of β was set to equal to the following expression:

$$\beta = \frac{1}{2} \left(\frac{1}{\rho_l} + \frac{1}{\rho_v} \right) \tag{3.71}$$

Therefore, the same value was adopted in this work.

3.6.2 Generic initial and boundary conditions

In numerical modeling, the initial conditions and the boundary conditions are of utmost importance to realistically characterizing any problem modeled by differential equations. Furthermore, in 3-dimensional simulations, where limitations on computer resources may be more restrictive, the domain length must be carefully chosen by taking into consideration the velocity field, as observed by [90] and [41]. A free-slip condition at the wall can be specified to avoid this effect whilst periodic boundary conditions may be used to avoid large domains and excessive computing time. The boundary conditions used in this work are briefly explained below, followed by their detailed specifications in each particular case in the results section:

- No-slip condition: all the velocity components are specified with null value, thus the fluid adjacent to the wall is at rest;
- Free-slip condition: used when a symmetry condition is desired. The normal velocity component to the wall is set to zero and the derivative of the tangent component is also set to null value, thus the adjacent fluid is in motion.
- Inflow condition: this is specified when an influx of mass is desired. For such a condition, $\mathbf{v} = \mathbf{v}_{inflow}$.
- Outflow condition: usually defined as $p = 0$ when the normal tension to the fluid outflow boundary is equal to zero. Such a condition represents a state where the fluid flow exits and its details are not known prior to solution of the flow problem or it is close to a fully developed condition.
- moving wall condition: a tangent velocity $\mathbf{v}_t \neq 0$ is set at the wall boundaries so that the reference frame is no longer fixed in the space, but instead it is moving with the same tangent velocity $\mathbf{v}_t$.
- symmetry condition: only the normal velocity $\mathbf{v}_n$ is prescribed on the wall boundary, thus a symmetry condition is imposing allowing the fluid next to the wall to slip in the tangent direction.

4 Finite Element Method

In this chapter, the Finite Element method formulation will be described. Focus will be on the variational formulation (weak form) of the non-dimensional conservation equations presented in Chapter 3. Additionally, the discrete finite elements used in this work will be presented and a brief discussion of the tetrahedron mesh generation method through the *Delaunay* algorithm. Finally, the methodology employed to the solution of the resulting linear system of equations will be presented.

4.1 Variational Formulation

The variational formulation will be presented for one set of equations, more specifically for the equations in which the Reynolds number Re is defined. The same formulation can be employed similarly to obtain their description when the Archimedes number N is defined.

Let consider the incompressible Navier-Stokes equation which comprises the momentum, continuity and energy (heat) equations in their non-dimensional form:

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{c} \cdot \nabla \mathbf{v} = -\frac{1}{\rho}\nabla p + \frac{1}{Re}\nabla \cdot [\mu(\nabla \mathbf{v} + \nabla \mathbf{v}^T)] + \frac{1}{Fr^2}\mathbf{g} + \frac{1}{We}\mathbf{f} \tag{4.1}$$

$$\nabla \cdot \mathbf{v} = 0 \tag{4.2}$$

$$\frac{\partial T}{\partial t} + \mathbf{c} \cdot \nabla T = \frac{1}{RePr}\nabla \cdot (k \nabla T) \tag{4.3}$$

valid in a domain $\Omega \subset \mathbb{R}^m$ with the following boundary conditions:

$$\mathbf{v} = \mathbf{v}_\Gamma \quad \text{in } \Gamma_1 \tag{4.4}$$

$$\mathbf{v}_t = 0 \quad \text{and } \sigma^{nn} = 0 \quad \text{in } \Gamma_2 \tag{4.5}$$

$$T = T_\Gamma \quad \text{in} \Gamma_3 \tag{4.6}$$

where the convective velocity **c** represents the relative velocity between the flow field and the mesh, given by the following expression: $\mathbf{c} = \mathbf{v} - \hat{\mathbf{v}}$. Here, **v** stands for the flow field velocity and $\hat{\mathbf{v}}$ for the mesh velocity.

Now, let us consider the subspace:

$$\mathbb{V} = H^1(\Omega)^m = \{\mathbf{v} = (v_1, \dots, v_m) : v_i \in H^1(\Omega), \forall\ i = 1, \dots, m\} \tag{4.7}$$

where $H^1(\Omega)$ and the *Sobolev* space given by:

$$H^1(\Omega) = \left\{ v \in L^2(\Omega) : \frac{\partial v}{\partial x_i} \in L^2(\Omega), i = 1, \cdots, m \right\} \tag{4.8}$$

taking $L^2(\Omega)$ as an infinite dimensional space characterized by the Lesbegue's integral, which is equivalent to the Riemann's integral for continuous functions, and thus it can be treated by the conventional form:

$$L^2(\Omega) = \left\{ v : \Omega \to \mathbb{R}, \int_\Omega v^2 d\Omega < \infty \right\} \tag{4.9}$$

Note that $\mathbb{V} = H^1(\Omega)^m$ is the Cartesian product of m spaces $H^1(\Omega)$ where:

$$\mathbb{V}_{\mathbf{v}_\Gamma} = \{\mathbf{v} \in \mathbb{V} : \mathbf{v} = \mathbf{v}_\Gamma \text{ in } \Gamma_1\} \tag{4.10}$$

$$\mathbb{P}_{p_\Gamma} = \{q \in L^2(\Omega) : q = p_\Gamma \text{ in } \Gamma_2\} \tag{4.11}$$

$$\mathbb{T}_{T_\Gamma} = \{r \in L^2(\Omega) : r = T_\Gamma \text{ in } \Gamma_3\} \tag{4.12}$$

The variational formulation consists in finding the solutions $\mathbf{v}(x,t) \in \mathbb{V}_{\mathbf{v}_\Gamma}$, $p(x,t) \in \mathbb{P}_0$ and $T(x,t) \in \mathbb{T}_{T_\Gamma}$ so that:

$$\int_\Omega \left\{ \frac{\partial \mathbf{v}}{\partial t} + \mathbf{c}\cdot\nabla\mathbf{v} + \frac{1}{\rho}\nabla p - \frac{1}{Re}\nabla\cdot[\mu(\nabla\mathbf{v}+\nabla\mathbf{v}^T)] - \frac{1}{Fr^2}\mathbf{g} - \frac{1}{We}\mathbf{f} \right\}\cdot\mathbf{w}\, d\Omega = 0 \tag{4.13}$$

$$\int_\Omega [\nabla\cdot\mathbf{v}]\, q\, d\Omega = 0 \tag{4.14}$$

$$\int_\Omega \left\{ \frac{\partial T}{\partial t} + \mathbf{c}\cdot\nabla T - \frac{1}{RePr}\nabla\cdot(k\nabla T) \right\} r\, d\Omega = 0 \tag{4.15}$$

Developing the equation terms, they become:

$$\begin{aligned} &\int_\Omega \left\{ \frac{\partial \mathbf{v}}{\partial t} + \mathbf{c}\cdot\nabla\mathbf{v} \right\}\cdot\mathbf{w}\, d\Omega + \int_\Omega \left\{ \frac{1}{\rho}\nabla p \right\}\cdot\mathbf{w}\, d\Omega \\ &\quad - \int_\Omega \left\{ \frac{1}{Re}\nabla\cdot[\mu(\nabla\mathbf{v}+\nabla\mathbf{v}^T)] \right\}\cdot\mathbf{w}\, d\Omega - \int_\Omega \left\{ \frac{1}{Fr^2}\mathbf{g} \right\}\cdot\mathbf{w}\, d\Omega - \int_\Omega \left\{ \frac{1}{We}\mathbf{f} \right\}\cdot\mathbf{w}\, d\Omega = 0 \end{aligned} \tag{4.16}$$

$$\int_\Omega \{\nabla\cdot\mathbf{v}\}\, q\, d\Omega = 0 \tag{4.17}$$

$$\int_\Omega \left\{ \frac{\partial T}{\partial t} + \mathbf{c}\cdot\nabla T \right\}\cdot r\, d\Omega - \int_\Omega \left\{ \frac{1}{RePr}\nabla\cdot(k\nabla T) \right\} r\, d\Omega = 0 \tag{4.18}$$

The first terms of Eqs. (4.16 and 4.18) will be treated in the *ALE* formulation as a substantive derivative for further development using by the semi-Lagrangian method. Its weighing ratio consists in:

$$\int_\Omega \frac{D\mathbf{v}}{Dt}\cdot \mathbf{w}d\Omega = \int_\Omega \frac{\partial \mathbf{v}}{\partial t} + \mathbf{c}\cdot\nabla\mathbf{v} \tag{4.19}$$

$$\int_\Omega \frac{DT}{Dt} r d\Omega = \int_\Omega \frac{\partial \mathbf{v}}{\partial t} + \mathbf{c}\cdot\nabla T \tag{4.20}$$

The next step is the treatment of the diffusive terms, in which the Green theorem should by applied, thus splitting the volume integral in two other integrals: the inner and the boundary domain integrals, so that:

$$\begin{aligned}\int_\Omega \nabla\cdot[\mu(\nabla\mathbf{v}+\nabla\mathbf{v}^T)]\cdot\mathbf{w}d\Omega = \\ -\int_\Omega \nu[(\nabla\mathbf{v}+\nabla\mathbf{v}^T):\nabla\mathbf{w}^T]d\Omega + \int_\Gamma \mathbf{n}\cdot[\mu(\nabla\mathbf{v}+\nabla\mathbf{v}^T)\cdot\mathbf{w}]d\Gamma\end{aligned} \tag{4.21}$$

$$\int_\Omega \nabla\cdot(k\nabla T) r d\Omega = -\int_\Omega (k\nabla T)\cdot\nabla r^T d\Omega + \int_\Gamma \mathbf{n}\cdot(k\nabla T) r d\Gamma \tag{4.22}$$

where the operator (:) stands for the scalar product between two tensors. The boundary integral Γ in the above equation may be segregated into two other integrals in Γ_1 and Γ_2. Due to $w = 0$ from Eq. (4.21) and $r = 0$ from Eq. (4.22), the integral in Γ_2 is null. Moreover, the integral in Γ_1 is also null due to the boundary conditions in (Eq. 4.5). Therefore the integral in Γ is null. The treatment of the pressure term is done by applying the same procedure, thus the integration by parts results in:

$$\int_\Omega \nabla p\cdot\mathbf{w}d\Omega = -\int_\Omega p\nabla\cdot\mathbf{w}d\Omega + \int_\Gamma p\mathbf{w}\cdot\mathbf{n}d\Gamma \tag{4.23}$$

where the above boundary integral is null due to the boundary conditions $\mathbf{w} = 0$ in Γ_1 and

$p = 0$ in Γ_2. The gravity and surface tension term are treated simply as:

$$\int_\Omega \mathbf{g} \cdot \mathbf{w} d\Omega \qquad \text{and} \qquad \int_\Omega \mathbf{f} \cdot \mathbf{w} d\Omega \tag{4.24}$$

The resulting equations are:

$$\int_\Omega \frac{D\mathbf{v}}{\partial t} \cdot \mathbf{w} d\Omega - \frac{1}{\rho}\int_\Omega p\nabla \cdot \mathbf{w} d\Omega + \frac{1}{Re}\int_\Omega \mu[\nabla\mathbf{v} + \nabla\mathbf{v}^T] : \mathbf{w} d\Omega - \frac{1}{Fr^2}\int_\Omega \mathbf{g} \cdot \mathbf{w} d\Omega - \frac{1}{We}\int_\Omega \mathbf{f} \cdot \mathbf{w} d\Omega = 0 \tag{4.25}$$

$$\int_\Omega [\nabla \cdot \mathbf{v}] q d\Omega = 0 \tag{4.26}$$

$$\int_\Omega \frac{DT}{\partial t} r d\Omega + \frac{1}{RePr}\int_\Omega (k\nabla T)\nabla r^T d\Omega = 0 \tag{4.27}$$

The integral forms are defined as follows:

$$m(\frac{D\mathbf{v}}{Dt}, \mathbf{w}) = \int_\Omega \frac{D\mathbf{v}}{Dt} \cdot \mathbf{w} d\Omega \tag{4.28}$$

$$k(\nu, \mathbf{v}, \mathbf{w}) = \int_\Omega \nu[(\nabla\mathbf{v} + \nabla\mathbf{v}^T) : \nabla\mathbf{w}^T] d\Omega \tag{4.29}$$

$$g(p, \mathbf{w}) = \int_\Omega \nabla p \cdot \mathbf{w}\, d\Omega \tag{4.30}$$

$$d(p, \mathbf{w}) = \int_\Omega (\nabla \cdot \mathbf{w}) p\, d\Omega \tag{4.31}$$

$$\tilde{k}(k, T, r) = \int_\Omega k\nabla T \cdot \nabla r^T d\Omega \tag{4.32}$$

$$\tilde{m}(\frac{DT}{Dt}, r) = \int_\Omega \frac{DT}{Dt} r d\Omega \tag{4.33}$$

Thus, the weak form of the proposed problem is written as: **Find the solutions $\mathbf{v}(x, t) \in \mathbb{V}_{\mathbf{v}_\Gamma}$,**

$p(x,t) \in \mathbb{P}$ **and** $T(x,t) \in \mathbb{T}_{T_\Gamma}$ **such that**

$$\begin{aligned} m(\frac{D\mathbf{v}}{Dt},\rho,\mathbf{w}) - g(p,\mathbf{w}) + \frac{1}{Re}k(\mu,\mathbf{v},\mathbf{w}) - \frac{1}{Fr^2}m(\mathbf{g},\rho,w) - \frac{1}{We}m(\mathbf{f},w) &= 0 && (4.34)\\ d(q,\mathbf{v}) &= 0 && (4.35)\\ \tilde{m}(\frac{DT}{Dt},r) + \frac{1}{RePr}\ \tilde{k}(k,T,r) &= 0 && (4.36) \end{aligned}$$

for all $\mathbf{w} \in \mathbb{V}_0$**,** $q \in \mathbb{P}_0$ **and** $T \in \mathbb{T}_0$.

4.2 The semi-discrete Galerkin Method

In this section, the semi-discrete Galerkin method for the governing equations will be formally presented. These equations are discretized only in the spatial domain, remaining continuous in the temporal domain.

So, let us consider the momentum equation in its variational non-dimensional form coupled in the orthogonal directions x, y and z:

$$\begin{aligned} &\int_\Omega \Big[\frac{Du}{Dt}w_x + \frac{Dv}{Dt}w_y + \frac{Dw}{Dt}w_z\Big]d\Omega - \frac{1}{\rho}\int_\Omega \Big[p\frac{\partial w_x}{\partial x} + p\frac{\partial w_y}{\partial y} + p\frac{\partial w_z}{\partial z}\Big] + \\ &\frac{1}{Re}\int_\Omega \mu\Big\{\Big(\frac{\partial u}{\partial x}\frac{\partial w_x}{\partial x} + \frac{\partial v}{\partial x}\frac{\partial w_y}{\partial x} + \frac{\partial w}{\partial x}\frac{\partial w_z}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial w_x}{\partial y} + \frac{\partial v}{\partial y}\frac{\partial w_y}{\partial y} + \frac{\partial w}{\partial y}\frac{\partial w_z}{\partial y} + \\ &\frac{\partial u}{\partial z}\frac{\partial w_x}{\partial z} + \frac{\partial v}{\partial z}\frac{\partial w_y}{\partial z} + \frac{\partial w}{\partial z}\frac{\partial w_z}{\partial z} + \Big) + \Big(\frac{\partial u}{\partial x}\frac{\partial w_x}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial w_y}{\partial x} + \frac{\partial u}{\partial z}\frac{\partial w_z}{\partial x} + \frac{\partial v}{\partial x}\frac{\partial w_x}{\partial y} + \\ &\frac{\partial v}{\partial y}\frac{\partial w_y}{\partial y} + \frac{\partial v}{\partial z}\frac{\partial w_z}{\partial y} + \frac{\partial w}{\partial x}\frac{\partial w_x}{\partial z} + \frac{\partial w}{\partial y}\frac{\partial w_y}{\partial z} + \frac{\partial w}{\partial z}\frac{\partial w_z}{\partial z}\Big) - \Big\}d\Omega \\ &\qquad \frac{1}{Fr^2}\int_\Omega (g_x w_x + g_y w_y + g_z w_z)d\Omega - \frac{1}{We}\int_\Omega (f_x w_x + f_y w_y + f_z w_z)d\Omega = 0 \qquad (4.37) \end{aligned}$$

According to the variational formulation, it is required to find the solutions $\mathbf{v} = (u,v,w) \in \mathbb{V}_{\mathbf{v}_\Gamma}$ and $p \in \mathbb{P}$ so that Eq. 4.37 must be true for all $\mathbf{w} = (w_x, w_y, w_z) \in \mathbb{V}_0$. Note that if the following expressions are satisfied:

$$
\begin{aligned}
&\int_\Omega \frac{Du}{Dt} w_x d\Omega - \frac{1}{\rho}\int_\Omega p\frac{\partial w_x}{\partial x} + \\
&\frac{1}{Re}\int_\Omega \mu\Big(\frac{\partial u}{\partial x}\frac{\partial w_x}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial w_x}{\partial y} + \frac{\partial u}{\partial z}\frac{\partial w_x}{\partial z} + \frac{\partial u}{\partial x}\frac{\partial w_x}{\partial x} + \frac{\partial v}{\partial x}\frac{\partial w_x}{\partial y} + \frac{\partial w}{\partial x}\frac{\partial w_x}{\partial z}\Big) d\Omega - \\
&\qquad\qquad\qquad \frac{1}{Fr^2}\int_\Omega g_x w_x d\Omega - \frac{1}{We}\int_\Omega f_x w_x d\Omega = 0
\end{aligned}
\tag{4.38}
$$

$$
\begin{aligned}
&\int_\Omega \frac{Dv}{Dt} w_y d\Omega - \frac{1}{\rho}\int_\Omega p\frac{\partial w_y}{\partial y} + \\
&\frac{1}{Re}\int_\Omega \mu\Big(\frac{\partial v}{\partial x}\frac{\partial w_y}{\partial x} + \frac{\partial v}{\partial y}\frac{\partial w_y}{\partial y} + \frac{\partial v}{\partial z}\frac{\partial w_y}{\partial z} + \frac{\partial u}{\partial y}\frac{\partial w_y}{\partial x} + \frac{\partial v}{\partial y}\frac{\partial w_y}{\partial y} + \frac{\partial w}{\partial y}\frac{\partial w_y}{\partial z}\Big) d\Omega - \\
&\qquad\qquad\qquad \frac{1}{Fr^2}\int_\Omega g_y w_y d\Omega - \frac{1}{We}\int_\Omega f_y w_y d\Omega = 0
\end{aligned}
\tag{4.39}
$$

$$
\begin{aligned}
&\int_\Omega \frac{Dw}{Dt} w_z d\Omega - \frac{1}{\rho}\int_\Omega p\frac{\partial w_z}{\partial z} + \\
&\frac{1}{Re}\int_\Omega \mu\Big(\frac{\partial w}{\partial x}\frac{\partial w_z}{\partial x} + \frac{\partial w}{\partial y}\frac{\partial w_z}{\partial y} + \frac{\partial w}{\partial z}\frac{\partial w_z}{\partial z} + \frac{\partial w}{\partial z}\frac{\partial w_z}{\partial x} + \frac{\partial w}{\partial z}\frac{\partial w_z}{\partial y} + \frac{\partial w}{\partial z}\frac{\partial w_z}{\partial z}\Big) d\Omega - \\
&\qquad\qquad\qquad \frac{1}{Fr^2}\int_\Omega g_z w_z d\Omega - \frac{1}{We}\int_\Omega f_z w_z d\Omega = 0
\end{aligned}
\tag{4.40}
$$

for any $w_x \in \mathbb{V}_0$, $w_y \in \mathbb{V}_0$ and $w_z \in \mathbb{W}_0$ respectively, then (Eq. 4.37) is automatically satisfied. Therefore, it is possible to solve such an equation by splitting it into the x-direction (Eq. 4.38), the y-direction (Eq. 4.39) and the z-direction (Eq. 4.40) separately, without any loss of generality. The continuity equations is then:

$$
\int_\Omega \Big(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\Big) q d\Omega = 0 \tag{4.41}
$$

and the energy equation:

$$
\int_\Omega \frac{DT}{Dt} r d\Omega - \frac{1}{RePr}\int_\Omega k\Big(\frac{\partial T}{\partial x}\frac{\partial r}{\partial x} + \frac{\partial T}{\partial y}\frac{\partial r}{\partial y} + \frac{\partial T}{\partial z}\frac{\partial r}{\partial z}\Big) d\Omega = 0 \tag{4.42}
$$

Considering NV the number of velocity nodes, NP the number of pressure nodes, NT the number of temperature nodes and NE the number of finite elements in the discretized domain Ω, the following substitutions in (Eq. 4.37) are required for the *Galerkin* method:

$$u(\mathbf{x},t) \approx \sum_{n=1}^{NV} N_n(\mathbf{x})u_n(t) \tag{4.43}$$

$$v(\mathbf{x},t) \approx \sum_{n=1}^{NV} N_n(\mathbf{x})v_n(t) \tag{4.44}$$

$$w(\mathbf{x},t) \approx \sum_{n=1}^{NV} N_n(\mathbf{x})w_n(t) \tag{4.45}$$

$$T(\mathbf{x},t) \approx \sum_{n=1}^{NT} T_n(\mathbf{x})c_n(t) \tag{4.46}$$

$$p(\mathbf{x},t) \approx \sum_{n=1}^{NP} P_n(\mathbf{x})p_r(t) \tag{4.47}$$

which are semi-discrete approximations, i.e. continuous in time (t) and discrete in space (x). Here, $N_n(x)$, $P_n(x)$ and $T_n(x)$ represent velocity, pressure and temperature interpolation functions, respectively.

In the momentum equation, the weight functions w_x, w_y e w_z are replaced by interpolation functions $N_m = N_m(x)$, $m = 1,\ldots,NV$. Therefore, making the suitable substitutions in Eqs. (4.38, 4.39 and 4.42), thus results in:

$$\begin{aligned}
&\sum_e \int_{\Omega^e} \sum_n \frac{Du_n}{Dt} N_m N_n d\Omega - \sum_e \int_{\Omega^e} \sum_r \frac{\partial N_m}{\partial x} P_r p_r d\Omega \\
&+ \frac{1}{Re} \sum_e \int_{\Omega^e} \sum_n \mu^e \Big(\frac{\partial N_m}{\partial x} \frac{\partial N_n}{\partial x} u_n + \frac{\partial N_m}{\partial y} \frac{\partial N_n}{\partial y} u_n + \\
&\frac{\partial N_m}{\partial z} \frac{\partial N_n}{\partial z} u_n + \frac{\partial N_m}{\partial x} \frac{\partial N_n}{\partial x} u_n + \frac{\partial N_m}{\partial y} \frac{\partial N_n}{\partial x} v_n + \frac{\partial N_m}{\partial z} \frac{\partial N_n}{\partial x} v_n \Big) d\Omega - \\
&\qquad \frac{1}{Fr^2} \sum_e \int_{\Omega^e} \sum_n N_m N_n g_{x,n} d\Omega - \frac{1}{We} \sum_e \int_{\Omega^e} \sum_n N_m N_n f_{x,n} = 0
\end{aligned} \tag{4.48}$$

$$\begin{aligned}
&\sum_e \int_{\Omega^e} \sum_n \frac{Dv_n}{Dt} N_m N_n d\Omega - \sum_e \int_{\Omega^e} \sum_r \frac{\partial N_m}{\partial y} P_r p_r d\Omega \\
&+ \frac{1}{Re} \sum_e \int_{\Omega^e} \sum_n \mu^e \Big(\frac{\partial N_m}{\partial x} \frac{\partial N_n}{\partial x} v_n + \frac{\partial N_m}{\partial y} \frac{\partial N_n}{\partial y} v_n + \\
&\frac{\partial N_m}{\partial z} \frac{\partial N_n}{\partial z} v_n + \frac{\partial N_m}{\partial x} \frac{\partial N_n}{\partial y} v_n + \frac{\partial N_m}{\partial y} \frac{\partial N_n}{\partial y} v_n + \frac{\partial N_m}{\partial z} \frac{\partial N_n}{\partial y} v_n \Big) d\Omega - \\
&\qquad \frac{1}{Fr^2} \sum_e \int_{\Omega^e} \sum_n N_m N_n g_{y,n} d\Omega - \frac{1}{We} \sum_e \int_{\Omega^e} \sum_n N_m N_n f_{y,n} = 0
\end{aligned} \tag{4.49}$$

$$\begin{aligned}
&\sum_e \int_{\Omega^e} \sum_n \frac{Dw_n}{Dt} N_m N_n d\Omega - \sum_e \int_{\Omega^e} \sum_r \frac{\partial N_m}{\partial z} P_r p_r d\Omega \\
&+ \frac{1}{Re} \sum_e \int_{\Omega^e} \sum_n \mu^e \Big(\frac{\partial N_m}{\partial x} \frac{\partial N_n}{\partial x} w_n + \frac{\partial N_m}{\partial y} \frac{\partial N_n}{\partial y} w_n + \\
&\frac{\partial N_m}{\partial z} \frac{\partial N_n}{\partial z} w_n + \frac{\partial N_m}{\partial x} \frac{\partial N_n}{\partial z} w_n + \frac{\partial N_m}{\partial y} \frac{\partial N_n}{\partial z} w_n + \frac{\partial N_m}{\partial z} \frac{\partial N_n}{\partial z} w_n \Big) d\Omega - \\
&\qquad \frac{1}{Fr^2} \sum_e \int_{\Omega^e} \sum_n N_m N_n g_{z,n} d\Omega - \frac{1}{We} \sum_e \int_{\Omega^e} \sum_n N_m N_n f_{z,n} = 0
\end{aligned} \tag{4.50}$$

The energy equation does not require high order elements since there is no coupling with pressure field. The unknown values are computed only at the tetrahedron vertices, thus pairing with the dimension of the unknown values of pressure $NT = NP$; however, for better understanding, the dimensions of unknown temperatures and pressures will be kept respectively as NT and NP, so that:

$$\begin{aligned}
&\sum_e \int_{\Omega^e} \sum_n \frac{DT_n}{Dt} T_m T_n d\Omega \\
&\qquad + \frac{1}{RePr} \sum_e \int_{\Omega^e} \sum_n k^e \Big(\frac{\partial T_m}{\partial x} \frac{\partial T_n}{\partial x} T_n + \frac{\partial T_m}{\partial y} \frac{\partial T_n}{\partial y} T_n + \frac{\partial T_m}{\partial z} \frac{\partial T_n}{\partial z} T_n \Big) d\Omega = 0
\end{aligned} \tag{4.51}$$

As mentioned at the beginning of this chapter, the continuity equation is strongly linked to the pressure, thus it may be evaluated at the pressure nodes, and therefore the weight function q is approximated by the interpolation functions associated to the pressure $P_r(x)$:

$$\sum_e \int_{\Omega^e} \sum_n \Big(\frac{\partial N_n}{\partial x} u_n + \frac{\partial N_n}{\partial y} v_n + \frac{\partial N_n}{\partial z} w_n \Big) P_r \; d\Omega = 0 \tag{4.52}$$

for $r = 1, \ldots, NP$. Restricting the interpolation functions to each element e, it follows that:

$$\sum_e \int_{\Omega^e} \sum_{j,k\in e} \Big(\frac{\partial N_j^e}{\partial x} u_j + \frac{\partial N_j^e}{\partial y} v_j + \frac{\partial N_j^e}{\partial z} w_j \Big) P_k^e \, d\Omega = 0 \tag{4.53}$$

Equations (4.48, 4.49, 4.50, 4.53 and 4.51) may be represented by the system of ordinary differential equations (ODE) below:

$$M_x \dot{u} + \frac{1}{Re}\{(2K_{xx} + K_{yy} + K_{zz})u + K_{xy} + K_{xz}w\} - G_x p - \frac{1}{Fr^2} M_{\rho,x} g_x - \frac{1}{We} M_x f_x \tag{4.54}$$

$$M_x \dot{v} + \frac{1}{Re}\{K_{yx}u + (K_{xx} + 2K_{yy} + K_{zz})v + K_{yz}w\} - G_y p - \frac{1}{Fr^2} M_{\rho,y} g_y - \frac{1}{We} M_y f_y \tag{4.55}$$

$$M_x \dot{w} + \frac{1}{Re}\{K_{zx}u + K_{zy}v + (K_{xx} + K_{yy} + 2K_{zz})w\} - G_w p - \frac{1}{Fr^2} M_{\rho,z} g_z - \frac{1}{We} M_z f_z \tag{4.56}$$

$$D_x u + D_y v + D_z w \tag{4.57}$$

$$M_T \dot{T} + \frac{1}{RePr}(K_{Txx} + K_{Tyy} + K_{Tzz})T \tag{4.58}$$

where $\dot{u}$, $\dot{v}$, $\dot{w}$ and $\dot{T}$ represent the substantial derivative and are defined as $\dot{u} = [Du_1/Dt, \ldots, Du_{NU}/Dt]^T$, $\dot{v} = [Dv_1/Dt, \ldots, Dv_{NU}/Dt]^T$, $\dot{w} = [Dw_1/Dt, \ldots, Dw_{NU}/Dt]^T$, $\dot{T} = [DT_1/Dt, \ldots, \partial T_{NT}/\partial t]^T$, $u = [u_1, \ldots, u_{NU}]^T$, $v = [v_1, \ldots, v_{NV}]^T$, $w = [w_1, \ldots, w_{NV}]^T$, $T = [T_1, \ldots, T_{NT}]^T$, $p = [p_1, \ldots, p_{NP}]^T$, which are the vectors of nodal values for the velocity, pressure and temperature, respectively. The respective matrix system associated to the ODE's are represented by:

$$\begin{array}{ccc}
M_x = \mathscr{A}_x(m^e), & M_y = \mathscr{A}_y(m^e), & M_z = \mathscr{A}_z(m^e), \\
K_{xx} = \mathscr{A}_x(k_{xx}^e), & K_{xy} = \mathscr{A}_x(k_{xy}^e), & K_{xz} = \mathscr{A}_x(k_{xz}^e), \\
K_{yx} = \mathscr{A}_y(k_{yx}^e), & K_{yy} = \mathscr{A}_y(k_{yy}^e), & K_{yz} = \mathscr{A}_y(k_{yz}^e), \\
K_{zx} = \mathscr{A}_z(k_{zx}^e), & K_{zy} = \mathscr{A}_z(k_{zy}^e), & K_{zz} = \mathscr{A}_z(k_{zz}^e), \\
G_x = \mathscr{A}_x(g_x^e), & G_y = \mathscr{A}_y(g_y^e), & G_z = \mathscr{A}_z(g_z^e), \\
D_x = \mathscr{A}_x(d_x^e), & D_y = \mathscr{A}_y(d_y^e), & D_z = \mathscr{A}_z(d_z^e), \\
K_{Txx} = \mathscr{A}_x(k_{Txx}^e), & K_{Tyy} = \mathscr{A}_y(k_{Tyy}^e), & K_{Tzz} = \mathscr{A}_y(k_{Tzz}^e), \\
 & M_T = \mathscr{A}_T(m^e) &
\end{array}$$

such that the sub-matrices, m^e, k^e_{xx}, k^e_{xy}, k^e_{xz}, k^e_{yx}, k^e_{yy}, k^e_{yz},k^e_{zx}, k^e_{zy}, k^e_{zz}, g^e_x, g^e_y, g^e_z, d^e_x, d^e_y, d^e_z, k^e_{Txx}, k^e_{Tyy} and k^e_{Tzz} are locally defined matrices to each element, so that:

$$m^e_{ij} = \int_{\Omega^e} N^e_i N^e_j \, d\Omega \quad k^e_{xx,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial x}\frac{\partial N^e_j}{\partial x}) d\Omega \tag{4.59}$$

$$k^e_{xy,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial y}\frac{\partial N^e_j}{\partial x}) d\Omega \quad k^e_{xz,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial z}\frac{\partial N^e_j}{\partial x}) d\Omega \tag{4.60}$$

$$k^e_{yx,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial x}\frac{\partial N^e_j}{\partial y}) d\Omega \quad k^e_{yy,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial y}\frac{\partial N^e_j}{\partial y}) d\Omega \tag{4.61}$$

$$k^e_{yz,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial z}\frac{\partial N^e_j}{\partial y}) d\Omega \quad k^e_{zx,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial x}\frac{\partial N^e_j}{\partial z}) d\Omega \tag{4.62}$$

$$k^e_{zy,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial y}\frac{\partial N^e_j}{\partial z}) d\Omega \quad k^e_{zz,ij} = \int_{\Omega^e} \nu^e (\frac{\partial N^e_i}{\partial z}\frac{\partial N^e_j}{\partial z}) d\Omega \tag{4.63}$$

$$g^e_{x,ik} = \int_{\Omega^e} \frac{\partial N^e_i}{\partial x} P^e_k \, d\Omega \quad g^e_{y,ik} = \int_{\Omega^e} \frac{\partial N^e_i}{\partial y} P^e_k \, d\Omega \tag{4.64}$$

$$g^e_{z,ik} = \int_{\Omega^e} \frac{\partial N^e_i}{\partial z} P^e_k \, d\Omega \quad d^e_{x,kj} = \int_{\Omega^e} \frac{\partial N^e_j}{\partial x} P^e_k \, d\Omega \tag{4.65}$$

$$d^e_{y,kj} = \int_{\Omega^e} \frac{\partial N^e_j}{\partial y} P^e_k \, d\Omega \quad d^e_{z,kj} = \int_{\Omega^e} \frac{\partial N^e_j}{\partial z} P^e_k \, d\Omega \tag{4.66}$$

$$m^e_{T,ij} = \int_{\Omega^e} N^e_i N^e_j \, d\Omega \quad k^e_{Txx,ij} = \int_{\Omega^e} k^e (\frac{\partial T^e_i}{\partial x}\frac{\partial T^e_j}{\partial x}) d\Omega \tag{4.67}$$

$$k^e_{Tyy,ij} = \int_{\Omega^e} k^e (\frac{\partial T^e_i}{\partial y}\frac{\partial T^e_j}{\partial y}) d\Omega \quad k^e_{Tzz,ij} = \int_{\Omega^e} k^e (\frac{\partial T^e_i}{\partial z}\frac{\partial T^e_j}{\partial z}) d\Omega \tag{4.68}$$

The operator $\mathscr{A}$ represents the assembling of elementary sub-matrices of the ODE's system ((Eq. 4.58), with respect to the global and local indexes of Eqs (4.48, 4.49, 4.50).

The dimension of the matrices in Eq. 4.58 are $NV \times NP$ for G_x, G_y and G_z, $NP \times NV$ for D_x, D_y and D_z and $NV \times NV$ for all the remaining matrices. From (Eq. 4.58) it is possible to rewrite the ODE's system in a compact way, coupling the velocities in x, y and z, resulting in:

$$\begin{aligned} M\dot{\mathbf{v}} + \frac{1}{Re} K\mathbf{v} - Gp - \frac{1}{Fr^2} M_\rho \mathbf{g} - \frac{1}{We} M\mathbf{f} &= 0 \\ D\mathbf{v} &= 0 \\ M_T \dot{T} + \frac{1}{RePr} K_T T &= 0 \end{aligned} \tag{4.69}$$

where the variables are now defined as $\dot{\mathbf{v}} = [D\mathbf{v}_1/\,Dt,\dots,D\mathbf{v}_{NV}/\,Dt,\mathbf{v}_1,\dots,\mathbf{v}_{NV}]^T$, $\mathbf{v} = [\mathbf{v}_1,\dots,\ \mathbf{v}_{NV},\mathbf{v}_1,\dots,\mathbf{v}_{NV}]^T$, $G = [g_1^x,\dots,g_{NV}^x,g_1^y,\dots,g_{NV}^y,g_1^z,\dots,g_{NV}^z]^T$, $D = [d_1^x,\dots,d_{NV}^x,d_1^y,\dots,d_{NV}^y,d_1^z,\dots,d_{NV}^z]^T$, $c = [c_1,\dots,\ c_{NT}]^T$, $\dot{c} = [DT/\,Dt,\dots,DT_{NT}/\,Dt,T,\dots,T_{NT}]^T$ and $p = [p_1,\dots,p_{NP}]^T$, and the matrices as

$$\mathbf{M} = \begin{bmatrix} M_x & 0 & 0 \\ 0 & M_y & 0 \\ 0 & 0 & M_z \end{bmatrix}_{3NV\times 3NV} \qquad \mathbf{M}_\rho = \begin{bmatrix} M_{\rho,x} & 0 & 0 \\ 0 & M_{\rho,y} & 0 \\ 0 & 0 & M_{\rho,z} \end{bmatrix}_{3NV\times 3NV}$$

$$\mathbf{K} = \begin{bmatrix} 2K_{xx} & K_{yx} & K_{zx} \\ K_{xy} & 2K_{yy} & K_{zy} \\ K_{xz} & K_{yz} & 2K_{zz} \end{bmatrix}_{3NV\times 3NV} \qquad \mathbf{G} = \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix}_{3NV\times NP}$$

$$\mathbf{D} = \begin{bmatrix} D_x D_y D_z \end{bmatrix}_{NP\times 3NV} \qquad \mathbf{M_T} = \begin{bmatrix} M_{Tx} & 0 & 0 \\ 0 & M_{Ty} & 0 \\ 0 & 0 & M_{Tz} \end{bmatrix}_{3NT\times 3NT}$$

$$\mathbf{K} = \begin{bmatrix} K_{Tx} & 0 & 0 \\ 0 & K_{Ty} & 0 \\ 0 & 0 & K_{Tz} \end{bmatrix}_{3NT\times 3NT}$$

4.3 The semi-Lagrangian Method

The semi-Lagrangian method has its own history dating from the end of the 1950's ([98]) and the beginning of the 1960's ([49] and [79]). However, the use of such a methodology for modeling fluid flow problems came later in the 1980's through the work of [78] and [73], in which predominately convective problems were investigated.

The semi-Lagrangian method was firstly used in convection-diffusion systems where numerical stability and large time steps were required. Since then, this methodology has been shown to fulfill the expectation for large problems to be solved relatively fast. For meteorological purposes, the numerical prediction of weather conditions using the semi-Lagrangian approach has shown to be very effective in which large time steps are crucial ([27]). However, the semi-Lagrangian method is not commonly found in the discretization of the Navier-Stokes equations, but recently work has shown its effectiveness for simulating flows with high velocities ([5] and [6]).

The semi-Lagrangian algorithm is an integrating factor method in which such a factor is an advection operator. This operator is shifted to a moving coordinate system from which the next time step quantity is calculated. Let ψ be a scalar function in its material derivative representation, given in 3-dimensional form as:

$$\frac{D\psi}{Dt} = \frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} + v\frac{\partial \psi}{\partial y} + w\frac{\partial \psi}{\partial w} \tag{4.70}$$

In the moving mesh context, the material derivative turns into a partial ordinary derivative, thus the convective term vanishes while the coordinates are instantaneously changed.

In fluid dynamics, the pure Lagrangian coordinates track the fluid flow by moving the points along its trajectory. However, this may limit the feasible range of simulation conditions, even for low velocity flows, since the particle's trajectory may turn to a chaotic state in a short period of time. Thus, to address such a drawback, the semi-Lagrangian method re-initializes the coordinate system at each time step, consequently recovering the original mesh. The quantity value in the current time step is calculated according to its position in the previous time step, thus this methodology is said to be explicit in time. However, due to its natural description, no time restriction is imposed to satisfy its numerical stability.

Following the description presented above, the Eq. (4.70) may be discretized linearly in the time domain at the point x_i by using a explicit first order scheme:

$$\frac{D\psi}{Dt} = \frac{\psi_i^{n+1} - \psi_d^n}{\Delta t} \tag{4.71}$$

where $\psi_d^n = \psi^n(x_d, t^n)$ and x_d is the departure point. In the strong form, the substantial derivative is calculated along the characteristic trajectory, thus finding the point x_d by solving $D\psi/Dt = f$ backwards in time $t^{n+1} \geq t \geq t^n$ using the initial condition $x(t^{n+1}) = x_i$. For simplicity, this procedure is shown in an 1-dimensional scheme in Fig. (4.1). According to this scheme, the initial position x_i in time t^{n+1} is known and therefore it is used to find the departure point x_d, whose position is unknown.

In the semi-Lagrangian context, a searching procedure is required to find the unknown departure points x_d in time t^n. This procedure may lead to excessive computational cost if it is not well designed, thus it should be treated with appropriate care. In this work, this searching procedure is implemented using a smart technique that maps each element node to the opposite element that shares the same face. Therefore, using volume coordinates, it is possible to track the path from the current node's position x_i to its departure point x_d as shown in Fig. (4.2). As can be seen, only few computational elements are required to find the element in which the departure point is located. Once the point position is known, an interpolation is performed and the quantity is assigned.

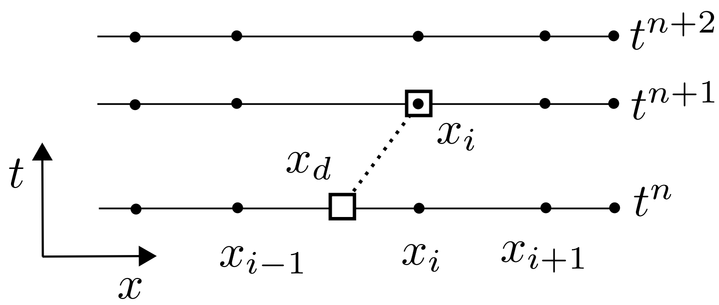

t^{n+2}
t^{n+1}
x_i
x_d
t
t^n
x_{i-1}
x_i
x_{i+1}
x

Figure 4.1: 1-dimensional space scheme of the semi-Lagrangian method. The point x_d is found by integrating the mesh backward in time according to Eq. (4.71). Thus, to calculate the quantity value, an interpolation is performed considering the quantity values on the nodes x_{i-1} and x_i.

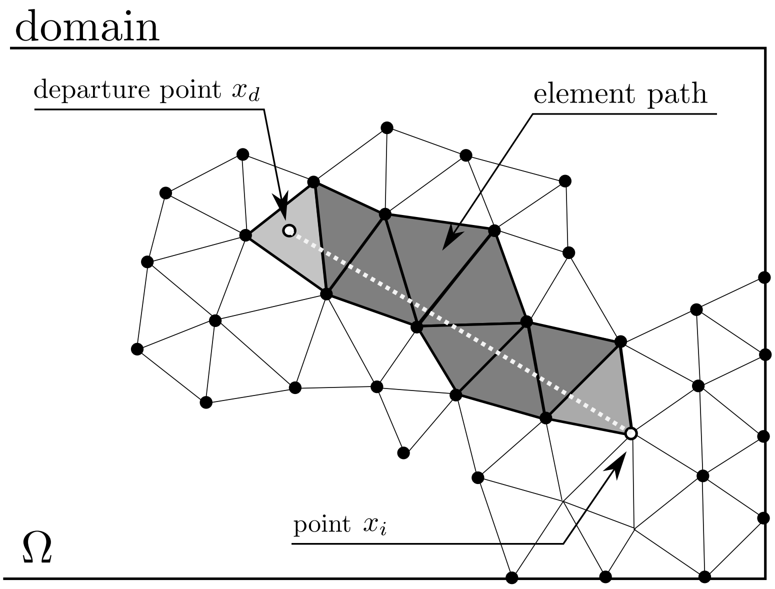

domain
departure point x_d
element path
point x_i
Ω

Figure 4.2: 2-dimensional particle trajectory from the current node's position ψ_i^{n+1} to its corresponding departure point ψ_d^n.

Besides the above example described in Fig. (4.2), many different others may appear. Figure (4.3) shows 5 possible trajectories that should be considered while developing the searching procedure. In the trajectory 1, the departure point x_d is locatead near the node position x_i. Once the point's position is found, the interpolated quantity can be assigned. In the trajectories 2 and 3, the departure points are located outside the boundary domain. Thus, a boundary element is used to interpolate the quantity to the nodes. In the trajectories 4 and 5, both cases are similar to the trajectory 4, however, the searching algorithm demands more computational time and the trajectory interpolation is less accurate.

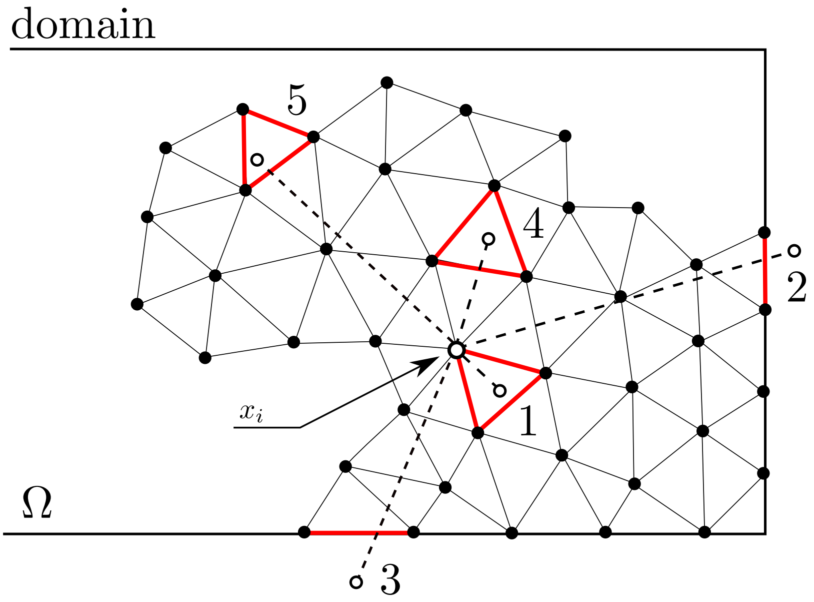


Figure 4.3: Interpolation procedures in the semi-Lagrangian method.

Above, a summary of the semi-Lagrangian method has been given. It is worth noting that higher accuracy may be achieved if the trajectory is fitted by high order schemes in time, such as the Generalized Adams-Bashforth schemes (see [75]). In the present work, the time restriction limits the displacement of the departure point to one element long, thus a linear approximation of the trajectory achieves reasonable accuracy.

4.3.1 Semi-Lagrangian Method for the Navier-Stokes Equations

The procedure described in the previous section will be used to discretize the material derivative of the *Navier-Stokes* equation. The material derivative (Eq. 4.72) is substituted into Eq. (4.34), resulting in:

$$\frac{D\mathbf{v}}{Dt} = \frac{\partial \mathbf{v}}{\partial t} + u - \hat{u}\frac{\partial \mathbf{v}}{\partial x} + v - \hat{v}\frac{\partial \mathbf{v}}{\partial y} + w - \hat{w}\frac{\partial \mathbf{v}}{\partial z} \tag{4.72}$$

$$\frac{\mathbf{v}_i^{n+1}-\mathbf{v}_d^n}{\Delta t} = -\frac{1}{\rho}\nabla p^{n+1} + \nabla\cdot[\mu(\nabla\mathbf{v}^{n+1}+\nabla(\mathbf{v}^{n+1})^T)] - \frac{1}{Fr^2}\mathbf{g}^n - \frac{1}{We}\mathbf{f}^n \tag{4.73}$$

Following the spirit of the semi-Lagrangian discretization method, Eqs. (4.34, 4.35 and 4.36) are re-written as:

$$\begin{aligned}
m(\frac{\mathbf{v}_i^{n+1}-\mathbf{v}_d^n}{\Delta t},\mathbf{w}) - g(p^{n+1},\mathbf{w}) + \frac{1}{Re}\ k(\mu,\mathbf{v}^{n+1},\mathbf{w}) - \frac{1}{Fr^2}(\mathbf{g},\mathbf{w}) - \frac{1}{We}(\mathbf{f},\mathbf{w}) &= 0 \qquad (4.74)\\
d(q,\mathbf{v}^{n+1}) &= 0, \qquad (4.75)\\
\tilde{m}(\frac{T_i^{n+1}-T_d^n}{\Delta t},r) + \frac{1}{RePr}\ \tilde{k_T}(k,T^{n+1},r) &= 0 \qquad (4.76)
\end{aligned}$$

for all $\mathbf{w}\in\mathbb{V}_0$, $q\in\mathbb{P}_0$ and $T\in\mathbb{T}_0$. Thus, the resulting discrete matrix form is given by:

$$\begin{aligned}
M\Big(\frac{\mathbf{v}_i^{n+1}-\mathbf{v}_d^n}{\Delta t}\Big) + \frac{1}{Re}K\mathbf{v}^{n+1} - Gp^{n+1} - \frac{1}{Fr^2}\mathbf{g}^n - \frac{1}{We}\mathbf{f}^n &= 0\\
D\mathbf{v}^{n+1} &= 0\\
M_T\Big(\frac{T_I^{n+1}-T_d^n}{\Delta t}\Big) + \frac{1}{RePr}K_T T^{n+1} &= 0
\end{aligned} \tag{4.77}$$

4.4 Mesh elements

The computational mesh used in the Finite Element Method allows for a wide variety of elements. They are characterized by their geometry and the polynomial interpolating function use to fit the desired equation. These elements may be classified according to their shape as triangular, quadrilateral, etc. for 2-dimensional problems and tetrahedral, prismatic, parallelepiped, etc. for 3-dimensional problems. Moreover, the order of the polynomial function may be classified as linear, quadratic, cubic, bi-linear, etc. Additionally, a particular class of elements namely *Isoparametric elements* may be used where the region to be modeled requires elements with more general shapes, which is the case of curvilinear domains (for details, see [103]). Even affordable to moving boundary problems, the Isoparametric elements are unfortunately relatively more complicated to deploy and, thus it has not been chosen in the present work.

The finite elements are numerically represented by the meaning of their local coordinates. This means that the interpolating functions (also known as shape functions) are built independently of the element's neighborhood, thus the finite element may be easily changed without modifying significantly the simulation program. Such an inherent ability is a strong advantage compared to other discretization methods such as finite difference and finite volume methods. In the following subsections, a description of the tetrahedron element through the *volume coordinates* and a brief illustration of different shape functions used in this work are presented.

4.4.1 Volume coordinates

Let us consider a point inside a generic tetrahedron $ijkl$ as depicted in Fig. (4.4). The local linear coordinates L_i, L_j, L_k and L_l may be calculated by its non-dimensional volume as follows:

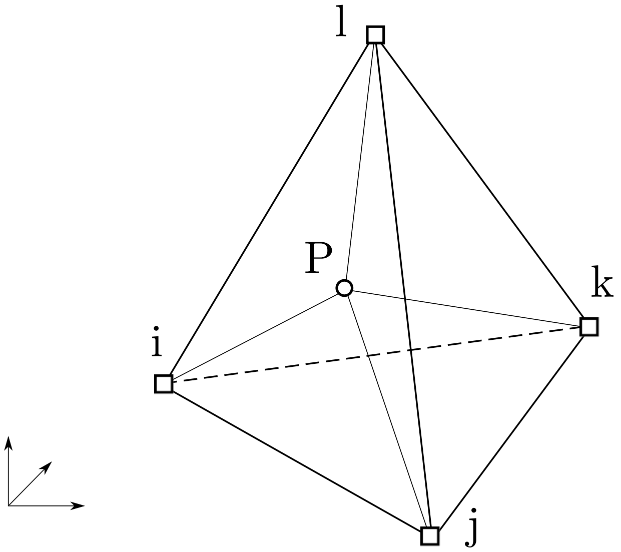


Figure 4.4: Volume coordinates for the generic tetrahedron i,j,k,l where P is a point somewhere inside the tetrahedron.

$$L_i = \frac{V_i}{V} \tag{4.78}$$

where V_i represents the volume occupied by the tetrahedron $ijkP$ and V the total volume of the tetrahedron element. Repeating the same idea for the remaining sides, it is possible to define L_j, L_k and L_l. Thus, the following is true for the tetrahedron element:

$$L_j = \frac{V_j}{V} \qquad L_k = \frac{V_k}{V} \qquad L_l = \frac{V_l}{V} \tag{4.79}$$

$$V_i + V_j + V_k + V_l = V \qquad \longrightarrow \qquad \frac{V_i}{V} + \frac{V_j}{V} + \frac{V_k}{V} + \frac{V_l}{V} = 1 \tag{4.80}$$

$$L_i + L_j + L_k + L_l = 1 \tag{4.81}$$

and re-writing as function of the local Cartesian coordinates x, y and z:

$$x = L_i x_i + L_j x_j + L_k x_k + L_l x_l \tag{4.82}$$

$$y = L_i y_i + L_j y_j + L_k y_k + L_l y_l \tag{4.83}$$

$$z = L_i z_i + L_j z_j + L_k z_k + L_l z_l \tag{4.84}$$

The element volume can be found by the matrix determinant, which is represented by the element coordinates:

$$6V = det \begin{bmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_4 \\ 1 & x_4 & y_4 & z_4 \end{bmatrix}$$

Solving the above system of Eqs. (4.82, 4.83 and 4.84), the linear coordinates L_1, L_2, L_3 and L_4 are equally determined and thus can be used to compute the high order elements:

$$L_1 = \frac{a_i + b_i x + c_i y + d_i z}{6V} \tag{4.85}$$

$$L_2 = \frac{a_j + b_j x + c_j y + d_j z}{6V} \tag{4.86}$$

$$L_3 = \frac{a_k + b_k x + c_k y + d_k z}{6V} \tag{4.87}$$

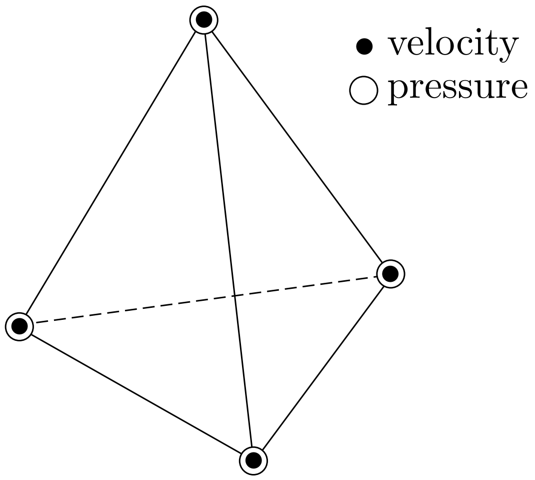
$$L_4 = \frac{a_m + b_m x + c_m y + d_m z}{6V} \tag{4.88}$$

4.4.2 3-dimensional elements

The choice of the appropriate element is a key step to successfully achieve the required precision in the simulations. In fluid dynamics, the element is responsible for the coupling of velocity and pressure and it should satisfy the requirements of the so-called *Ladyzhenskaya-Babouska-Brezzi* (LBB) stability condition ([21], [102] and [67]). Such a condition imposes the type of velocity and pressure basis functions. One way to avoid the LBB condition is to use stabilizing methods such as pressure stabilization, penalty method or artificial compressibility ([44], [103],[102]). However, such a stabilization process is not part of this work, in which has preferred the use of LBB stable elements.

The elements used in the present work are reported below, followed by short descriptions of their features and their applicability in the Navier-Stokes equation.

Linear element: The unknowns are evaluated at the tetrahedron's corners with interpolation functions of order 1. This element is commonly used to solve scalar equations, such as heat and chemical species transport. This element does not satisfy the LBB condition for fluid flow problems and it cannot be used to solve velocity and pressure without stabilizing methods.



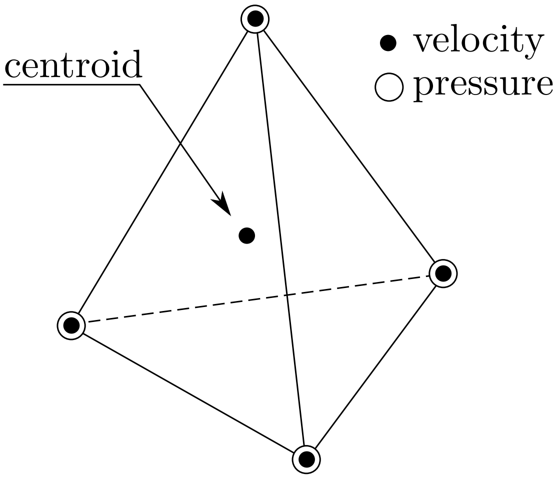
For linear tetrahedron elements with local coordinates L_i, L_j, L_k and L_l, the shape functions are equivalent and written as:

$$N_i = L_i, \qquad i = 1,2,3,4 \tag{4.89}$$

A simple formula is available in the literature to calculate the volume integral over the element (see [43], [103]) and it is expressed by:

$$\iiint_V {L_i}^a {L_j}^b {L_k}^c {L_m}^d \, dx dy dz = \frac{a! b! c! d!}{(a+b+c+d+3)!} 2V \tag{4.90}$$

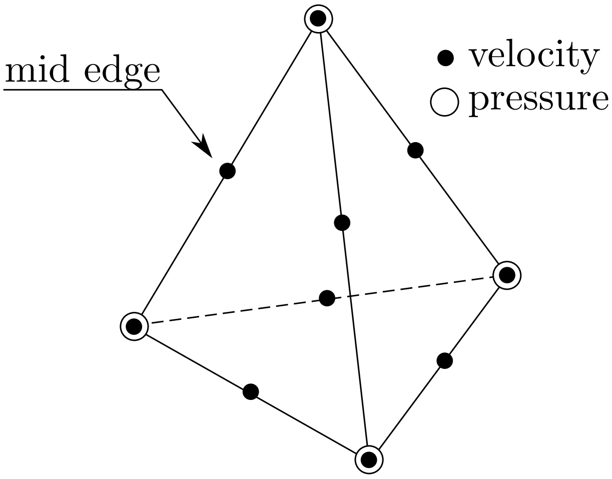
Mini element: This element is part of the Taylor-Hood family and it is a combination of the linear tetrahedron and an additional “bubble” function, built by an additional node localized in its barycenter, thus making it a five node element. The interpolation polynomial is of order 4, but it does not present the second and the third degree terms. The pressure is linear and is evaluated at the tetrahedron vertices and the velocity is cubic incomplete and evaluated at all the 5 nodes.



The shape functions N_i for the 5-node tetrahedron element may be defined as a function of local coordinates L_i, L_j, L_k as follows:

$$\begin{aligned} N_i &= L_i - 64 L_1 L_2 L_3 L_4, \qquad i = 1,2,3,4 \\ N_5 &= 256 L_1 L_2 L_3 L_4 \end{aligned} \tag{4.91}$$

10-node element: Defined by nodes in the middle of the tetrahedron edges, the 10-nodes quadratic element is commonly used in fluid flow problems. The interpolation polynomial is of order 2. The pressure is linear and evaluated at the tetrahedron vertices, while the velocity is quadratic and evaluated at all 10 nodes.



The interpolation functions N_i for the 10-node quadratic element may be also expressed as a function of local coordinates L_i, L_j, L_k as follows:

$$\begin{aligned} N_i &= (2L_i - 1)L_i, \qquad i = 1,2,3,4 \\ N_5 &= 4L_1L_2 \\ N_6 &= 4L_2L_3 \\ N_7 &= 4L_1L_3 \\ N_8 &= 4L_1L_4 \\ N_9 &= 4L_2L_4 \\ N_{10} &= 4L_3L_4 \end{aligned} \tag{4.92}$$

The list of available elements are not limited to those above described. Many other geometries and polynomial functions may be used to discretize the fluid flow equations. High order elements, discontinous pressure elements and combined elements are examples of the wide variety of Finite Elements. This diversity is organized by families such as Taylor-Hood, Crouzeix-Raviart and Serendipity; each one has its particular approach to calculate the quantities. A list of available elements for fluid flow problems can be found at [52], [44] and [103].

Due to its superior mass conservation and the minimum amount of nodes per element to satisfy the LBB condition, the mini-element has been chosen to discretize pressure and velocity in the the Navier-Stokes equations within the moving mesh context. Such a decision has shown to be appropriate to model two-phase flow problems with minimum implementation efforts and significant accuracy.

4.5 The Delaunay Tetrahedralization

Tetrahedron elements are extremely powerful to discretize any kind of geometry. Its application extends from simple brick geometries to more complex shapes including curvilinear boundaries. However, for fluid flow problems, the distribution of the elements should respect some physical requirements. While considering a domain discretized by tetrahedron elements and not limiting it to a uniform point distribution, an unstructured grid is expected, i.e. the number of a node's neighbors is not constant. Such a flexibility cannot be assessed if cubic and brick shaped elements are used, however the Finite Element method allows the concurrent use of different element shapes, if the code implementation takes into account their interconnectivity. Thus, the tetrahedron distribution in the computational domain may change with respect to the investigated problem and some criteria should be imposed to keep the elements bounded to acceptable aspect ratios, since low quality elements may corrupt the accuracy of the computational discretization.

In the present technique, the equations are discretized over an unstructured non-regular tetrahedral mesh with optimal element properties. This is achieved using the *Delaunay* tetrahedralization algorithm which ensures well-shaped elements. However, due to the constant motion of the mesh nodes, all the Delaunay requirements may not be satisfied. Nevertheless, efforts have been made to maintain the mesh quality, in each time step, restricting it to good element aspect ratios.

The Delaunay tetrahedralization consists of a geometrical construction often used in mesh generation for the Finite Element Method. Such a construction corresponds to the dual graph of the *Voronoi diagram* ([96]), which is a special type of space decomposition determined by distances and commonly used for mapping regions and land areas. Figure. (4.5) shows a simplified representation of the Voronoi diagram of a set of points and its corresponding tessellation according to the Delaunay properties.

In 2-dimensional spaces, the Delaunay algorithm maximizes the minimum angle of all triangles and minimizes the largest circumscribed circle, thus avoiding low quality elements. The Delaunay tetrahedralization follows the same strategy as its correspondent 2-dimensional algorithm, however some of the features may not be assured due to geometrical restrictions. Although this inconvenient, the Delaunay tetrahedralization results in satisfactory elements and thus can be used to discretize the domain of two-phase flow problems by the Finite Element Method (see [81] and [69]).

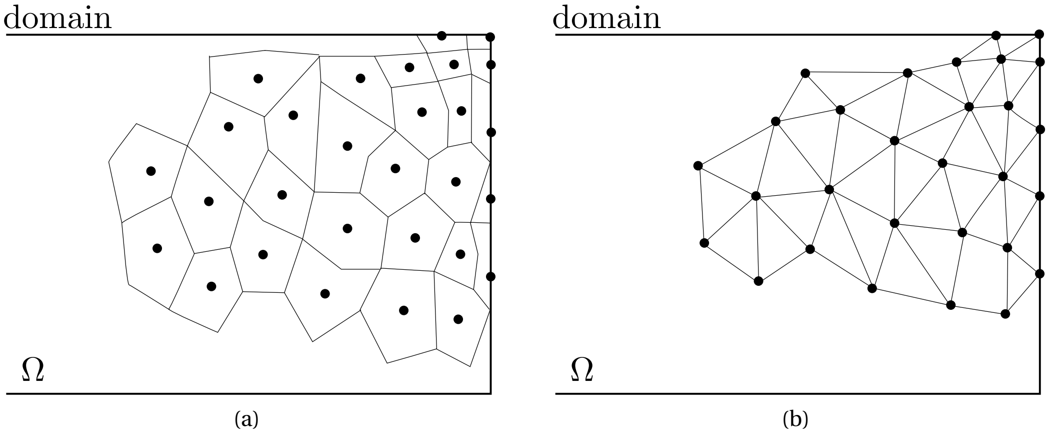


Figure 4.5: 2-dimensional representation of a (a) *Voronoi diagram* and (b) its tessellation according to the Delaunay properties.

4.6 Projection Method

After the spatial and temporal discretization by the Finite Element Method, the solution of a set of linear algebraic equations is required. There are many ways to solve this linear system, but due to its size, computational methods may be employed to avoid excessive processing time and memory overhead. The methods for the solution of a linear system may be divided in two groups, namely coupled and uncoupled. The former attempts to solve the complete linear system directly at each time step. However, the direct solution of the Navier-Stokes equations, where pressure and velocity are strongly connected, becomes an onerous procedure. Moreover, the non-linearity of such an equation makes the solution even more complex. Instead, the latter detaches the internal dependencies, allowing a serial resolution of the problem. As part of the uncoupled methods, the Projection Method has become popular in the fluid dynamics domain. It was originally introduced by Chorin ([20]) and thus followed by many others such as ([71]) with the SIMPLE method and [39] with the MAC method.

The family of Projection methods is basically divided into three different parts:

- Continuous projection method
- Semi-discrete projection method
- Discrete projection method

The projection method in the continuous form attempts to split velocity and pressure before the discretization of the equations is achieved. In the semi-discrete form, the procedure is applied when part of the Navier-Stokes's equation is already discretized in the time domain. The discrete projection method segregates pressure and velocity before the solution of the linear system, thus time and space are already discretized. This work has chosen a variant of the discrete projection method, namely LU-based discrete projection method. The general idea of the Projection method as well as the implemented discrete method is presented below.

4.6.1 The Projection Method

The projection method assumes that any vector $\mathbf{v} \in \Omega$, where Ω is a domain with a smooth boundary $\partial\Omega$, may be uniquely decomposed by:

$$\mathbf{v} = \mathbf{v}_d + \nabla\varrho \tag{4.93}$$

and $\mathbf{v}_d$ is solenoidal and parallel to the boundary $\partial\Omega$, that is,

$$\nabla \cdot \mathbf{v}_d = 0 \quad \text{in} \quad \Omega \tag{4.94}$$

$$\mathbf{v}_d \cdot \mathbf{n} = 0 \quad \text{in} \quad \partial\Omega \tag{4.95}$$

where ϱ is a scalar field. From the vector analysis, $\nabla \times \nabla\varrho = 0$, and thus Eq. 4.93 is equivalent to spliting the vector $\mathbf{v}$ in two components, namely divergence-free and rotational-free. Now, let us consider the equations in the conservative form valid for all the domain Ω with constant viscosity μ and density ρ in each phase:

$$\frac{\partial \mathbf{v}}{\partial t} + \nabla \cdot (\mathbf{v}\mathbf{v}) = -\nabla p + \nu\nabla^2\mathbf{v} + \mathbf{g} \tag{4.96}$$

$$\nabla \cdot \mathbf{v} = 0 \tag{4.97}$$

Thus, Eq. (4.96) is written as:

$$\frac{\partial \mathbf{v}}{\partial t} + \nabla p = \mathbf{S}(\mathbf{v}) \quad \text{where} \quad \mathbf{S}(\mathbf{v}) = \nu\nabla^2\mathbf{v} + \mathbf{g} - \nabla \cdot (\mathbf{v}\mathbf{v}) \tag{4.98}$$

In general, $\mathbf{S}(\mathbf{v})$ is not divergence-free and rotational-free. Additionally, note that:

$$\nabla \cdot \big[\frac{\partial \mathbf{v}}{\partial t} = \frac{\partial}{\partial t}(\nabla \cdot \mathbf{v}) = 0 \tag{4.99}$$

$$\nabla \times \nabla p = 0 \tag{4.100}$$

Equation (4.98), as suggested by [20], may be interpreted with **v** from Eq. (4.93). The vector $\mathbf{S}(\mathbf{v})$ is known and can be projected onto both divergence-free ($\partial \mathbf{v}/\partial t$) and rotational free ($\nabla p$) subspaces, in other words:

$$\frac{\partial \mathbf{v}}{\partial t} = \mathbf{P}[\mathbf{S}(\mathbf{v})] \qquad \nabla p = \mathbf{Q}[\mathbf{S}(\mathbf{v})] \tag{4.101}$$

where **P** and **Q** are projection operators which satisfy the following properties:

$$\mathbf{P}^2 = \mathbf{P} \quad \mathbf{Q}^2 = \mathbf{Q} \quad \mathbf{PQ} = \mathbf{QP} = 0 \tag{4.102}$$

Now, for any vector **v**, **P** projects such a vector onto the null space of the divergent operator and **Q** is reciprocal to the rotational operator, that is:

$$\nabla \cdot \mathbf{P}[\mathbf{v}] = 0 \qquad \forall \mathbf{v} \in \Omega \tag{4.103}$$

$$\nabla \cdot \mathbf{Q}[\mathbf{v}] = 0 \qquad \forall \mathbf{v} \in \Omega \tag{4.104}$$

By comparing Eqs. (4.99 and 4.101), the projection operators are:

$$\mathbf{P} = \mathbf{I} - \nabla(\nabla^2)^{-1}(\nabla) \tag{4.105}$$

$$\mathbf{Q} = \mathbf{I} - \mathbf{P} \tag{4.106}$$

As observed by Gresho [38] and due to the strong coupling of pressure and velocity in incompressible flows, only acceleration and pressure can be split by the projection operators.

4.6.2 The LU-based discrete projection method

The discrete projection method based on the *LU* decomposition is obtained through the factorization of the resulting linear system, so that the velocity and pressure splitting is done after the space and the time discretization:

$$M(\frac{\mathbf{v}^{n+1}-\mathbf{v}^{n}}{\Delta t})+\frac{1}{Re}K\mathbf{v}^{n+1}-Gp^{n+1}-\frac{1}{Fr^2}\mathbf{g}^{n}-\frac{1}{We}\mathbf{f}^{n}=0 \tag{4.107}$$

$$D\mathbf{v}^{n+1}=0 \tag{4.108}$$

$$M_T(\frac{T_i^{n+1}-T_d^{n}}{\Delta t})+\frac{1}{RePr}KT^{n+1}=0 \tag{4.109}$$

Equation (4.109) can be solved separately, however Eqs. (4.107 and 4.108) form a system of equations that may be represented by:

$$\begin{bmatrix} B & -\Delta tG \\ D & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} r^{n} \\ 0 \end{bmatrix} + \begin{bmatrix} bc_1 \\ bc_2 \end{bmatrix} \tag{4.110}$$

where the system is now written only for the problem unknowns such that $\mathbf{v}^{n+1}=[u_1^{n+1},\dots,u_{Nu}^{n+1},v_1^{n+1},\dots,v_{Nv}^{n+1},w_1^{n+1},\dots,w_{Nv}^{n+1}]^T$, $p^{n+1}=[p_1^{n+1},\dots,p_{Np}^{n+1}]^T$, where Nu, Nv, Nw e Np are the number of unknowns for velocity in x, y and z and pressure respectively. Note that the vector and matrix notation were kept as simple as possible. The **B** matrix is given by:

$$B=\frac{M_\rho}{\Delta t}+\frac{K}{Re} \tag{4.111}$$

and the right hand side vector stands for the known quantities in current time n,

$$r^n = \frac{M_\rho}{\Delta t}\mathbf{v}_d^n \tag{4.112}$$

and the boundary conditions, which are the contributions of known values for velocity and pressure on the right hand side of the system. The projection method based on the LU factorization aims to uncoupled the linear system (Eq. 4.110) through a block factorization. According to [51], such a procedure can be done in several different ways, each one classified by families of methods. Now, consider a *LU* canonical block factorization as:

$$\begin{bmatrix} B & -\Delta t G \\ D & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{B} & 0 \\ \mathbf{D} & \Delta t \mathbf{D}\mathbf{B_1^{-1}}\mathbf{G} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & -\Delta t \mathbf{B_2^{-1}}\mathbf{G} \\ 0 & \mathbf{I} \end{bmatrix} \tag{4.113}$$

Thus, the resulting system is described as:

$$\begin{bmatrix} \mathbf{B} & 0 \\ \mathbf{D} & \Delta t \mathbf{D}\mathbf{B_1^{-1}}\mathbf{G} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & -\Delta t \mathbf{B_2^{-1}}\mathbf{G} \\ 0 & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} r^n \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{bc_1} \\ \mathbf{bc_2} \end{bmatrix} \tag{4.114}$$

The *Uzawa* method [16] is achieved if the system presented in (Eq. 4.114) is solved. However its solution requires the computation of the inverse of matrix **B** at each time step, which is a time consuming task and extremely costly. Therefore, such a matrix may be approximated by a diagonal matrix $\mathbf{B_L}$ which is easy and fast to invert. This methodology is known as *Matrix Lumping* and it has shown to be suitable for the Finite Element Method. The main idea of this method is to perform a summation of all entries of a row into its main diagonal. Two diagonalizations were tested, one using the consistent mass matrix **M** so that the new approximated diagonal matrix results in $\mathbf{M_L}$, and the second using the matrix **B** and resulting in $\mathbf{B_L}$. Both approaches have shown good results, but a small oscillation was noted for the velocity field when the *Reynolds* number varies from low to moderate values. It is important to note that according to [16], different approximations can be done for the consistent matrices $\mathbf{B_1}$ and $\mathbf{B_2}$, but to satisfy the continuity equation, it is mandatory that $\mathbf{B_1} = \mathbf{B_2}$, so that all the approximation error is focused to the momentum equation. Thus, the new uncoupled linear system is described as follows:

$$\begin{bmatrix} \mathbf{B} & 0 \\ \mathbf{D} & \Delta t \mathbf{D}\mathbf{B_1^{-1}}\mathbf{G} \end{bmatrix} \cdot \begin{bmatrix} \tilde{\mathbf{v}}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} r^n \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{bc_1} \\ \mathbf{bc_2} \end{bmatrix} \tag{4.115}$$

$$\mathbf{B}\tilde{\mathbf{v}} = \mathbf{r^n} + \mathbf{bc_1} \tag{4.116}$$

$$\Delta t \mathbf{D}\mathbf{B^{-1}}\mathbf{G} p^{n+1} = -\mathbf{D}\tilde{\mathbf{v}} + \mathbf{bc_2} \tag{4.117}$$

$$\begin{bmatrix} \mathbf{I} & -\Delta t \mathbf{B_2^{-1}}\mathbf{G} \\ 0 & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{v}}^{n+1} \\ p^{n+1} \end{bmatrix} \tag{4.118}$$

$$\tag{4.119}$$

$$\mathbf{v}^{n+1} = \tilde{\mathbf{v}} + \Delta t \mathbf{B^{-1}}\mathbf{G} p^{n+1} \tag{4.120}$$

where t is time and $\mathbf{v}_d^n$ is the velocity calculated in the previous time step at the departure points. Thus, the complete solution procedure is described as follows:

- compute the trial velocity $\tilde{\mathbf{v}}$ by solving:

$$\mathbf{B}\tilde{\mathbf{v}} = \mathbf{r^n} + \mathbf{bc_1} \tag{4.121}$$

- update $\tilde{\mathbf{v}}$ while considering the surface and gravity forces:

$$\tilde{\mathbf{v}}_{corr} = \tilde{\mathbf{v}} + \Delta t \mathbf{M}_{\rho L}^{-1}(\frac{1}{Fr^2}\mathbf{M}_\rho \mathbf{g} + \frac{1}{We}\mathbf{M}\mathbf{f}) \tag{4.122}$$

- compute p^{n+1} from:

$$\mathbf{D}\mathbf{B^{-1}}\mathbf{G} p^{n+1} = -\mathbf{D}(\tilde{\mathbf{v}}_{corr}) + \mathbf{bc_2} \tag{4.123}$$

- find the solution velocity $\mathbf{v}^{n+1}$ from:

$$\mathbf{v}^{n+1} = \tilde{\mathbf{v}}_{corr} + \mathbf{B_L^{-1}G}p^{n+1} \tag{4.124}$$

The subscript L refers to the *Lumped* matrix, which consists in a diagonal approximation, thus avoiding the computation of the matrix inverse. Note that in Eq. (4.122) the source terms $\mathbf{f}$ and $\mathbf{g}$ are only included once the trial velocity $\tilde{\mathbf{v}}$ is computed, thus satisfying the balance with pressure. For consistency, the lumped approximation is also employed in the computation of $\mathbf{f}$, therefore $\mathbf{Mf} \approx \kappa \mathbf{G}H$.

The two final linear systems (Eqs. (4.121) and (4.123)) are solved at each time step by the conjugate gradient method and the generalized minimum residual method, respectively. The Incomplete Cholesky and LU factorization are used as preconditioners to speed up the convergence of the final solution.

4.6.3 Time step restriction

In the ALE formulation, the convection term is driven by the difference between two velocities, namely $\mathbf{u}$ and $\hat{\mathbf{u}}$. As explained in chapter 3), $\mathbf{u}$ is the total Eulerian velocity and $\hat{\mathbf{u}}$ is the mesh velocity. Instead of applying the classical Galerkin method, which imposes a time step and mesh space restriction, the semi-Lagrangian technique was chosen to discretize this term. This approach leads to an explicit modeling of the operator, and due to its character, there is no strict time step restriction for its stability. Nevertheless, large time steps should be avoided since the accuracy is first order in time, otherwise important non-linear phenomena may not be captured. The time step constraint for this term is given by the expression below:

$$\Delta t_{sl} < \frac{h_{min}}{\mathbf{u}_{max}} \tag{4.125}$$

where h_{min} is the smallest tetrahedral mesh edge length and $\mathbf{u}_{max}$ is the maximum value of velocity given by $\max\{|u|,|v|,|w|\}$. On the other hand, the motion of the mesh nodes is due to $\hat{\mathbf{u}}$ and the time step constraint should prevent the nodes from moving more than one mesh cell, otherwise the mesh is corrupted due to element distortions and collision of nodes. The strategy of the moving mesh time constrain calculation is based on the velocity differences and the local mesh edge size, thus limiting one node to overlap the other or cross the face of an element. Let us consider v_1 and v_2 as the vertices of a tetrahedral edge h_e and $\Delta\hat{\mathbf{u}}_e = \hat{\mathbf{u}}_1 - \hat{\mathbf{u}}_2$ as the mesh velocity difference, thus the proposed Lagrangian time step should be bounded by:

$$\Delta t_l < \min\Big(\frac{h_e}{|\Delta \hat{\mathbf{u}}_e|}\Big) \tag{4.126}$$

where, $\hat{\mathbf{u}}$ is the velocity of the moving mesh given by Eq. (6.5). Unfortunately, the Lagrangian time step may fairly restricts the time step, but using the node deletion method previously described, the mesh edge length h_e does not become too short and consequently the time step is increased. Additionally, the explicit treatment of gravity and surface tension adds a constraint to the final time step which is related to the wave velocity propagating into the computational mesh. According to Brackbill and Kothe [12] and Fortuna [30], such a criteria for both surface tension and gravity may be written as:

$$\Delta t_s < \Big[\frac{\rho h^3 \mathrm{Eo}}{2\pi}\Big]^{1/2} \tag{4.127}$$

and

$$\Delta t_g < \Big[\frac{1}{h_{min}}\Big]^{1/2} \tag{4.128}$$

In the above equations, ρ is an average fluid density between the inner and outer fluids, h is the mesh characteristic length, h_{min} is the smallest mesh length and Eo is the *Eötvös*. Thus, the final simulation constraint may be written as:

$$\Delta t \leq \min\Big\{\Delta t_l, \Delta t_s, \Delta t_g, \Delta t_{sl}\Big\} \tag{4.129}$$

4.6.4 Object-oriented design

The layout of a project is of utmost importance in many designing aspects and it should be treated with care. Reusability and further development must be considered when creating a project. Not different is the design of a numerical code, especially if it involves large data structures and the discretization of complex equations, which is the case of the present work. Moreover, the Finite Element method allows a modular structure that may be designed to provide easy maintenance and inclusion of different methods without the need to rewrite the entire code.

The *C++* language was chosen as the code programming language due to its many available features. This modern computer language has shown to be efficient in many applications such as systems software, application softwares, device drivers, high-performance computing, etc. Moreover, its high portability makes the *C++* language easily to be compiled in many different operating systems, such as UNIX-based sytems (Linux and OS X) and Microsoft Windows. Finally, the paradigm of object-oriented programming can be exhaustively used since this language allows the manipulation of objects, classes, hierarchy and polymorphism. Figure (4.6) shows a simplified Unified Model Diagram (UML) of the proposed project layout and its description is given below.

The *Model* class is responsible to characterize the conditions and the geometry of the proposed study, being the first class to be initialized. The non-dimensionalization of the domain, the boundary conditions, the insertion and deletion of nodes in the volumetric and surface meshes, the measurement of mesh quality and many other methods are implemented in this class. Moreover, since the methodology used in this work to compute the curvature is based on mesh geometric objects such as tangent and normal vectors, the curvature calculation is performed in the *Model* class.

The *MeshSmooth* class, as its name suggests, performs the mesh smoothing according to some parameters such as velocities, coordinates and edge lengths. This class is linked to the *Model* class and to the *Simulator* class, therefore modifying the computational mesh without insertion and deletion of nodes.

The volumetric points distribution is achieved by the *Helmholtz* class. The discretization of the Helmholtz equation in 3-dimensions is done using the finite element method, thus the code structure is reused as input parameters. In this case, the surface mesh is passed as an argument from the *Model* class and the finite linear element (*FEMLinElement*) is used to discretize the equation. The solution is than sent back to the *Model* class.

In the *TElement* class, the finite elements can be chosen according to their type. The design of such a class is based on hierarchy, which allows other classes to be attached, and consequently inherit the elemental array structures. Three additional classes are inherited from the *TElement* class, namely: *FEMLinElement*, *FEMMiniElement* and *FEMQuadElement*. Each of these classes are responsible to build the elemental array structure according to the element order. The elemental stiffness, mass, gradient and divergence matrices are created with the aid of the Gaussian quadrature which is computed in such a way that the accuracy of the elemental integrals are kept high. Note that the proposed modular design allows the inclusion of many different elements without affecting the other classes, thus not requiring any additional modification in the computational code.

The core of this numerical code is located in the *Simulator* class. There, the global matrices are assembled according to the elemental arrays from the *TElement* class and the domain mesh from *Model* class. The *Simulator* class is also responsible to impose the boundary conditions to the linear system, to assemble the source vectors of gravity and surface tension and to

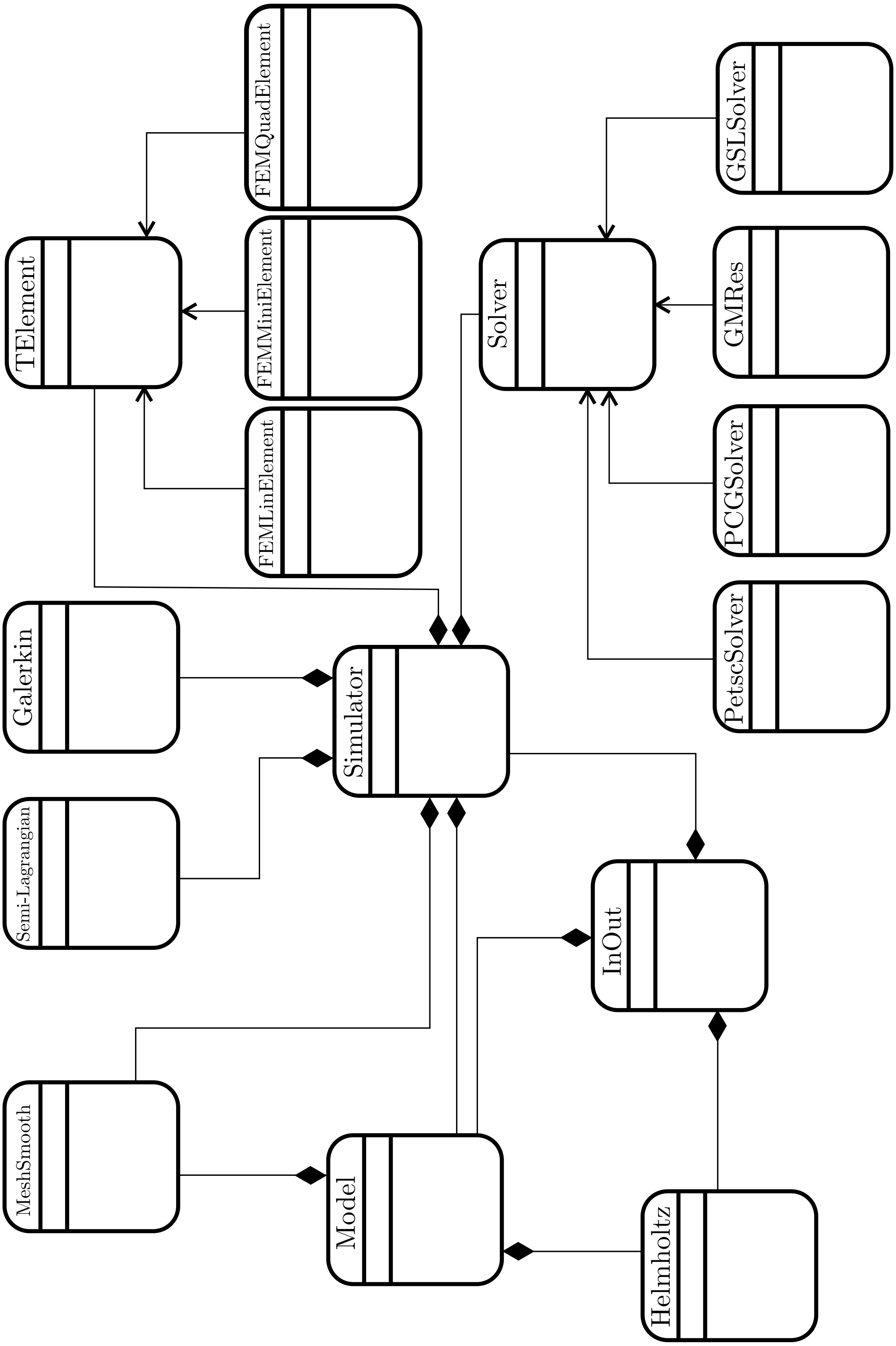

FEMQuadElement
GSLSolver
TElement
FEMMiniElement
Solver
GMRes
FEMLinElement
PCGSolver
Galerkin
Simulator
PetscSolver
Semi-Lagrangian
InOut
MeshSmooth
Model
Helmholtz

Figure 4.6: Simplified UML class diagram of the code's design. The classes are projected to allow Reusability and further development.

compute the moving mesh velocity.

The convective term of Navier-Stokes equations is computed in a modular way. Two classes were designed apart of the *Simulator* class, namely *Semi-Lagrangian* and *Galerkin* classes. The former computes the convection term using the semi-Lagrangian method and the later uses the Galerkin method. Note that for stability reasons, the *Semi-Lagrangian* class has been chosen for all the simulations here presented. Moreover, additional methods can easily be included into the code without affecting the structure of the *Simulator* class.

Finally, the assembled linear system with proper boundary conditions and source vectors are thus sent to the *Solver* class in which a suitable method will be chosen to compute the solution for the current time step. Among the implemented inheritor classes, one has been built as an interface to a popular scientific computation toolkit, namely *Petsc* ([7], [7], [8]). The class *PetscSolver* allows the use of any linear solver and preconditioners implemented by the *Petsc* group such as Conjugated Gradients, GMRes, Least Square Residual, , LU, Cholesky, Jacobi and many others. Additionally, the remaining inheritor classes of the *Solver* class are developed to solve the linear system by the Conjugated Gradients and GMRes methods. This extended flexibility brings into the code a potential platform for the study of efficient ways to solve large linear systems.

5 The Discrete Interface

This chapter describes the interface discretization procedure, from its geometrical representation to the nodal surface tension calculation. The Heaviside function is presented and the distribution of fluid properties in the numerical domain is compared to standard methods found in the literature.

5.1 Geometrical representation

In front-tracking codes the interface is constructed by a set of geometrical objects, such as triangles, edges and nodes, which are moved in Lagrangian fashion, whlie instead the background mesh is fixed on the space. An additional function is required to communicate from one mesh to another, since there is no implicit interconnectivity. This approach leads to the so-called zero-thickness interface, in which a sharp representation is achieved. Although its excellent geometrical definition, the fluid properties close to the interface require a numerical treatment to avoid undesirable instabilities. Thus, these properties are smoothed along the transition zone and the zero-thickness is no longer guaranteed.

Unlike front-tracking codes, the present surface and background meshes are part of the same computational mesh, thus no additional equation is required to pass the information from one mesh to another. The 3-dimensional mesh comprises a set of tetrahedron elements distributed on the domain and the interface is found by a scalar function, namely Heaviside, which defines the nodes belonging to each phase and the interface itself. To achieve a zero-thickness representation, the interface's nodes must be connected consistently so that the piecewise discrete interface may be represented by a set of interconnected triangles. In other words, each triangle is a face of two adjacent tetrahedral elements. Figure (5.1a) shows a schematic representation of the discrete interface between the two different phases. The same triangle face is shared by two adjacent tetrahedrons, therefore the zero-thickness interface is successfully achieved.

Advantages and drawbacks are found in such an approach, but one feature is especially inter-

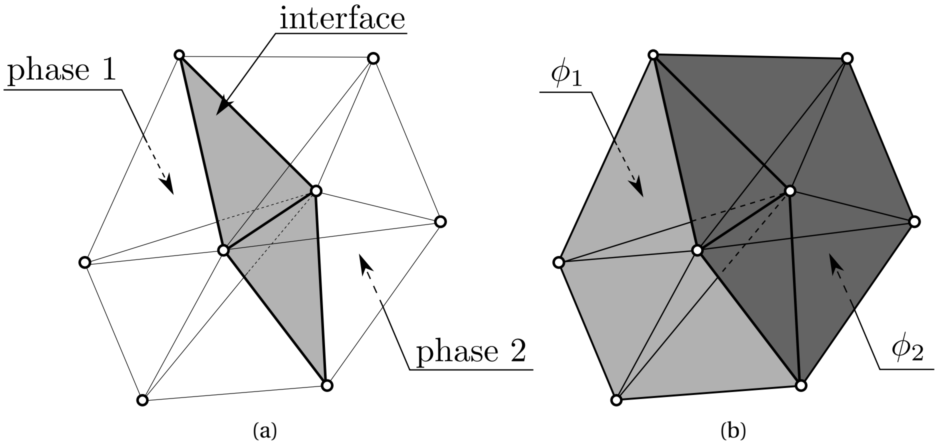
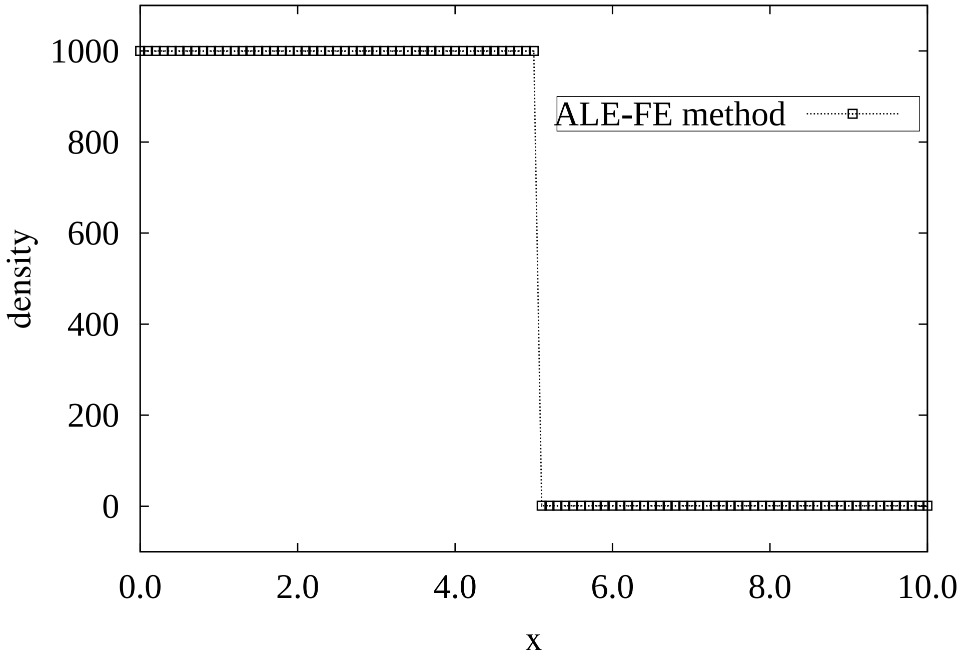


Figure 5.1: Geometrical representation of the interface between the phases. (a) The interface (gray colored) is represented by a set of triangles, edges and nodes which are part of the tetrahedron mesh. (b) The fluid property ϕ, such as density or viscosity, is sharply defined in phase 1 and phase 2 with a zero thickness interface in the transition zone.

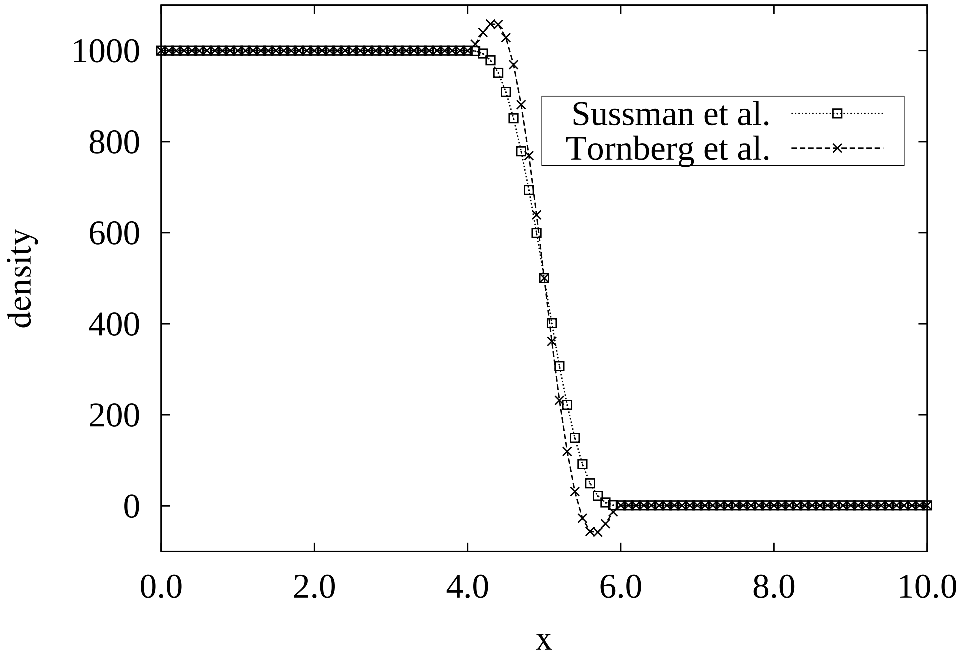
esting due to the definition of the fluid properties in the transition area. From the macroscopic point of view, the physical meaning of an interface is the region that sharply divides the volume occupied by each phase. Thus, it is desirable that such an interface's thickness should be kept as thin as possible. The Lagrangian description guaranties the geometrical part, but due to the abrupt change in properties from one phase to another, numerical instabilities may appear and deteriorate the accuracy of the solution. Such a problem is mainly due to the location of the interface somewhere in-between two computational elements. This can be circumvented with the ALE and the Finite Element formulation, in which the interface is not located in between mesh elements but it shares the faces of two adjacent computational mesh elements and thus the fluid properties remain constant in each mesh element. A sharp transition of properties is thus successfully achieved and does not require the use of any smoothing functions, consequently assuring accuracy in the balance of forces close to the interface.

Figure (5.1b) shows the transition zone between the two phases colored by dark and light gray, which was purposely drawn to highlight the methodology proposed by this work. As can be seen, the property ϕ_1 fills the elements of phase 1 and the property ϕ_2 fills exactly the elements of phase 2. Even for a high property ratio $\phi_1/\phi_2 = 1000$, the methodology proposed here does not present spurious oscillation in the pressure and the velocity field. Figure (5.2) depicts a 1-dimensional plot of density distribution along the phases. Figure (5.2a) shows a the density distribution used in the implemented ALE-FE scheme. Due to the Finite Element formulation, each phase property ϕ is assigned to each tetrahedron element, thus a sharp transition of properties is achieved. Figure (5.2b) shows a smoothed distribution commonly found in Level-Set methods. Such a procedure is required to avoid numerical instabilities close to the interface.

Despite the sharp definition of the interface and the fluid properties, topological changes are not naturally handled in this method, thus requiring an implementation effort on the modeling of coalescence and break-up of bubbles and drops. To address this issue, a geometrical model may be used to merge or split two surfaces. For instance, when the film thickness between two bubbles are smaller than a predefined parameter, the surfaces are connected and coalescence takes place. Although the topological change occurs, the physical aspects are not fulfilled. In fact, the mechanisms of bubble coalescence and break-up is still an open issue and, consequently, a potential field of future research.



(a)



(b)

Figure 5.2: Density distribution in two-phase flows. Phase 1 has a density $\rho_1 = 1000$ and phase 2 has a density $\rho_2 = 1$. The interface is at $x \approx 0.5$. (a) The sharp transition is achieved by the ALE-FE method, in which no artificial smoothing is required. (b) Smoothed distribution of density commonly found in Level-Set methods (see [87] and [93]).

5.2 Curvature and normal vectors in $\mathbb{R}^3$

According to Eq. (3.19), the non-dimensional form of the surface tension term can be written as $\mathbf{f} = \mathbf{n}\kappa\delta$, where κ is the curvature and $\mathbf{n}$ is the surface unity outward normal vector. Additionally, δ represents the Dirac delta function with support on the interface. Now, let us consider a distributed surface tension force scheme based on the Heaviside function:

$$\mathbf{f} = \sigma\kappa\nabla H \tag{5.1}$$

In such a scheme, the distributed interface force $\mathbf{f}$ is a volume force and its intensity $\sigma\kappa$ is calculated and applied on the direction of the gradient of the linear Heaviside function ∇H, where σ stands for the surface tension coefficient. Thus, at the interface all the surface tension force is well distributed on the free node's neighbors and the effects of overshooting and undershooting are eliminated.

In 2-dimensional space, the mean curvature κ can be locally calculated from the variation of the normal or tangent vector along the curve that defines the interface. Such a definition is derived from the *Frenet*'s formulas ([48]) and it is written as:

$$\kappa\mathbf{t} = -\frac{\partial\mathbf{n}}{\partial s} \qquad \text{or} \qquad \kappa\mathbf{n} = \frac{\partial\mathbf{t}}{\partial s} \tag{5.2}$$

which represents the continuous description. The discrete first order approximation is then given by:

$$\kappa\mathbf{n} = \frac{(\mathbf{t}_1 - \mathbf{t}_2)}{\Delta s} \tag{5.3}$$

where $\mathbf{t}_1$ and $\mathbf{t}_2$ are approximations of the unit tangent vectors in two consecutive interface edges and Δs is its approximate arc length. Figure (5.3a) shows the continuous representation of the mentioned formula and in Fig. (5.3b) the discrete representation of the 2-dimensional interface is presented. The interface's nodes are used to compute the tangent and normal vector and the approximate arc length is found by connecting the centroids of each segment.

Note that with the above scheme, the Continuous Surface Force (CSF) in the Eq. (3.18) can be

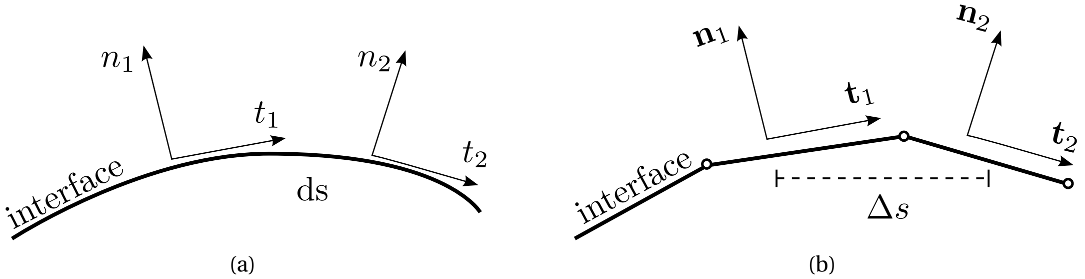


Figure 5.3: The 2-dimensional *Frenet*'s formula for mean curvature ([48]). (a) The continuous description and (b) the discrete form used in the computational grid.

written in the discrete form using Eqs. (5.1) and (5.3) as:

$$\mathbf{f} = -\sigma\left[\frac{(\mathbf{t}_1 - \mathbf{t}_2)}{\Delta s}\right] \cdot \mathbf{n}\nabla H \tag{5.4}$$

In the above equation, the capillary force intensity $\sigma|\mathbf{t}_1 - \mathbf{t}_2|$ is divided by the approximate arc length, that is applied in the direction of the gradient of the Heaviside function ∇H. In this scheme, the vector **n** can be determined in several ways. For instance, Level-Set methods compute the normal vector through the following scheme:

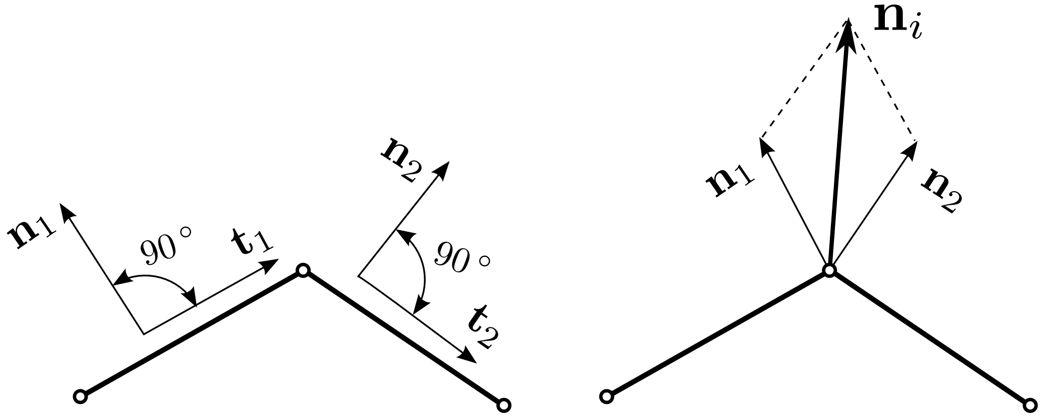
$$\mathbf{n} = -\frac{\nabla H}{|\nabla H|} \tag{5.5}$$

which gives a good approximation in the discrete space. The final discrete form of the surface tension can be computed as:

$$\mathbf{f} = \sigma\left[\frac{(\mathbf{t}_1 - \mathbf{t}_2)}{\Delta s}\right] \cdot \frac{\nabla H}{|\nabla H|}\nabla H \tag{5.6}$$

If the interface representation is explicitly defined by nodes, a more accurate evaluation of the normal vector **n** can be achived by orthogonalizing the same calculated unit tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ as shown in Fig. (5.4a). In 2-dimensional space, such a calculation is done relatively easy, since the interface mesh is always structured and thus the number of neighbors of each interface node is constant and equal to 2. The mean direction of $\mathbf{n}_i$ is a simple sum of unit normal vectors $\mathbf{n}_1$ and $\mathbf{n}_2$.

These schemes are compatible to a linear continuous approximation of the pressure finite element space and, consequently, can be successfully applied to the Navier-Stokes equation.



(a) (b)

Figure 5.4: Normal vector evaluation in 2-dimensional spaces. (a) The normal vector of each edge may be found by rotating the previously calculated tangent vector by 90°. (b) The final nodal normal vector **n** is found by summing the two normal vectors n_1 and n_2.

In this work, the equations are evaluated in $\mathbb{R}^3$ and the aforementioned scheme is no longer suitable. A new scheme is therefore proposed to compute the mean curvature and the normal vectors in 3-dimensional spaces, which will later be used in Eq. (5.1) to calculate the surface tension force. As explained earlier in this chapter, the interface between the fluids is represented by a set of geometrical objects which defines a surface, hence the new scheme should take into consideration the different topologies that such a surface may present. This scheme is described as follows:

Let Ω_s be an embedded surface in $\mathbb{R}^3$, n_i be the set of nodes n lying on the surface Ω_s and e_i^j be the set of surface triangles associated to the ith node, where j is the number of triangles, which may vary according to the structure of the surface mesh (Fig. 5.5). The surface tension force **f** requires the calculation of the mean curvature κ which, in the present model, is defined at each node i. An evaluation of this nodal mean curvature κ_i is done by integrating the elemental force contribution over the 1-ring triangle neighbors and dividing by the corresponding barycentric area. To calculate the elemental force, one should find two unit normal vectors, which lie on the triangle surface, and integrate them on the segments d_1 and d_2 as shown in Fig. (5.6a). The segments d_1 and d_2 connect the triangle mid-edges to the triangle centroid. Due to the Stokes theorem, the distributed elemental force $\mathbf{t}_n d$ in Fig. (5.6a) is equivalent to the integral of $\mathbf{t}_n \cdot \mathbf{n}$ over $d_1 + d_2$. Thus, a simplified way to calculate the elemental force is achieved by orthogonalizing one of the two vectors ($\mathbf{t}_1$ or $\mathbf{t}_2$ - Fig. 5.6b), finding its unit vector, and integrating the result to the segment d, which connects two triangle mid-edges (Fig. 5.6c). Consequently, an evalutation of the ith node mean curvature is found by integrating the intensity of elemental force $\mathbf{t}_\mathbf{n} d$ in the 1-ring triangle neighbors e_j^i and dividing it by the sum of the barycentric areas of e_j^i. The mentioned expression is calculated as follows:

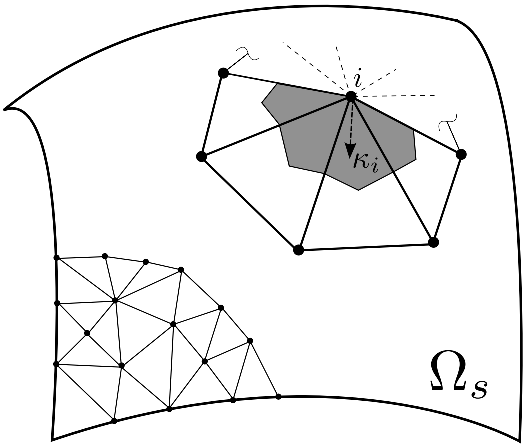
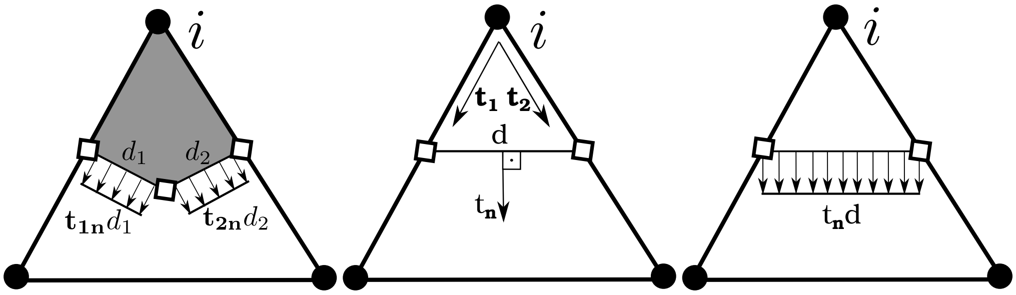


Figure 5.5: Schematic representation of the curvature evaluation κ_i at a common surface node, which is calculated using geometric operations at all the triangular neighboring elements and weighted by the barycentric area (gray colored).

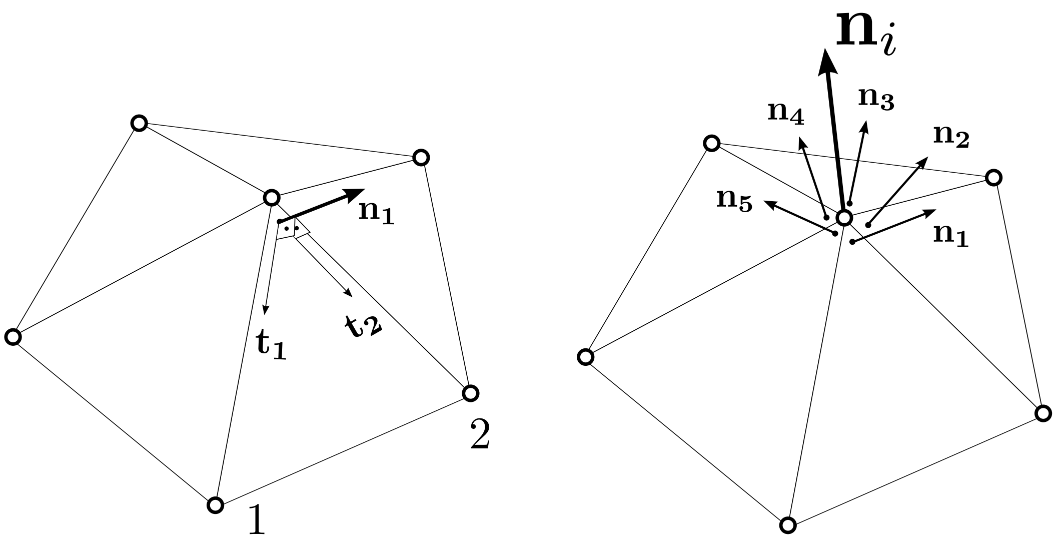


(a) (b) (c)

Figure 5.6: (a) An elemental force evaluation is done using the sum of the distributed forces $\mathbf{t_{1n}}d_1$ and $\mathbf{t_{2n}}d_2$. (b) Using the Stokes theorem, the elemental distributed force may be calculated orthogonalizing one of the two linearly independent vectors t_1 and t_2 to the segment d which connects two mid-edge nodes. (c) An evaluation of the node mean curvature is found by dividing the sum of the module of the calculated distributed forces ($|t_n d|$) by the sum of the barycentric areas (Eq. 5.7.)

$$\kappa_i = \frac{\left|\sum_{j=1}^{m} (\mathbf{t_n} d)_j\right|}{\sum_{j=1}^{m} A_j^i} \tag{5.7}$$

In the above equation, κ_i is the node mean curvature and A_j^i is the barycentric area of the jth triangle neighbor of i, which is equivalent to 1/3 of the triangle area, and m is the number of neighbor elements of i. The direction of application of such a curvature is given by the normal vector, that unfortunately cannot always be defined by the direction of the vector k_i itself due to a singular case where the set of triangular faces e_i^j are on the same plane, thus the length of the computed vector is equal to zero and its direction is then not defined. In this singular case, to find the normal vector of the node n_i one can approximate it by the sum of the cross product of two vectors of e_i^j since the 1-ring neighbor nodes are consistently sorted. This scheme is an extension for surfaces embedded in $\mathbb{R}^3$ of the previously presented 2-dimensional scheme shown in Fig. (5.7).



(a) (b)

Figure 5.7: Normal vector evaluation in 3-dimensional spaces. (a) The normal vector of each triangle in the surface may be found by applying the cross product of two tangent vectors which lie in the triangle plane. (b) The final nodal normal vector $\mathbf{n}_i$ is found by summing the normal vectors n_e for $e = \{1..j\}$. In the illustrated case $j = 5$.

Due to the unstructured character of the surface mesh, the number of neighbor triangular elements e associated to the node i must be taken into account, since it may vary from one node to another. Moreover, a consistent surface orientation is required in this scheme so that it can be successfully applied. Substituting such a distributed force into Eq. (5.1) yields:

$$\mathbf{f} = \sigma \frac{\left| \sum_{j=1}^{m} (\mathbf{t_n} d)_j \right|}{\sum_{j=1}^{m} A_j^i} \mathbf{n} \nabla H \tag{5.8}$$

the calculation of the surface tension force in 3-dimensions is thus achieved and thus can now be substituted into Eq. (3.32) as a source term.

5.3 The discrete surface tension force

As seen earlier in this chapter, the calculation of the surface tension force is based on the gradient of a Heaviside function ∇H. According to Chapter (4), a Finite Element formulation for the given force is achieved by considering the following scheme:

$$\frac{1}{We}\mathbf{M}\mathbf{f} = \frac{1}{We}\Sigma\mathbf{G}H_\lambda \tag{5.9}$$

In the above equation, Σ is a diagonal matrix with elements $\sigma\kappa_1, \sigma\kappa_2, \sigma\kappa_3, \cdots, \sigma\kappa_{NV}$, where NV is the total number of mesh nodes relative to the pressure field. The matrix **G** stands for the discrete form of the gradient operator ∇ and H_λ is the discrete Heaviside function. Finally, Equation (5.9) can be substituted into the discrete momentum equation (Eq. (4.69)) and the surface tension term calculation achieved.

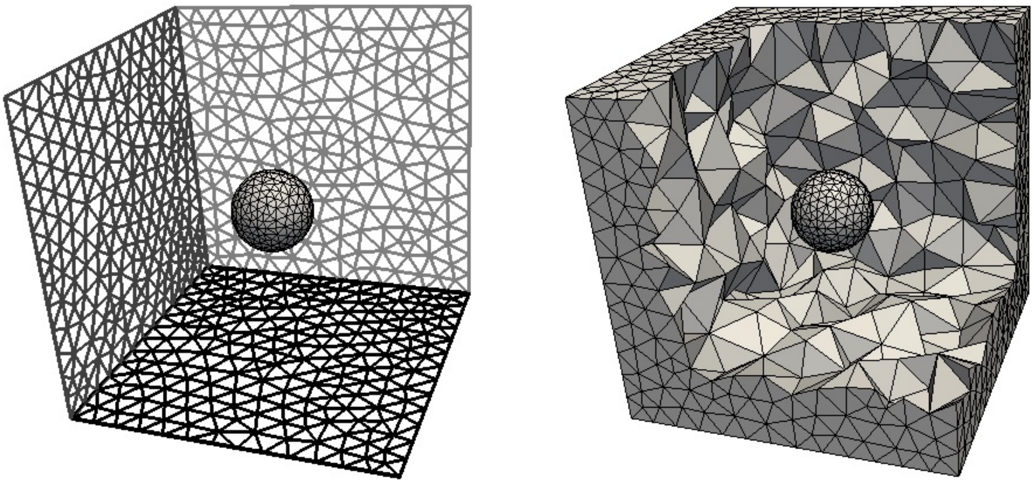
6 Adaptive mesh refinement

This chapter describes the new methodology of adaptive mesh refinement developed to work with the Finite Element method in two-phase flows, in which the interface between the fluids plays an important role. The effective management of the computational mesh is detailed in which two data structures are stored in the computer memory and handled separately.

6.1 Mesh representation

Two sets of data are stored during the simulation which are treated separately in the context of the adaptive mesh refinement: the volumetric nodes and the surface mesh. The latter consists of two parts, the interface between the phases and the domain boundary, both created simultaneously by the software (`GMesh`, [32]). The code is then linked to a tetrahedral mesh generator (`TETGEN`, [82]) that uses the previous generated surface meshes and a volumetric set of nodes as input parameters. The initial volumetric node distribution may be set manually, according to the requirements of the simulation, or by `TETGEN`, which may create a smooth distribution of nodes according to the edge lengths of the given surface meshes. Then, the 3-dimensional connectivity array is exported to the code. Figure (6.1) depicts the two data structures used in this work. Such an approach allows an easy maintenance of each region without affecting the node connectivity of the other. Moreover, the data structure makes the mesh management easier in the code programming level, enabling fast access to each structure separately.

In the ALE context the computational mesh is not fixed in the space, instead, it is moved according to an arbitrary velocity. Therefore to avoid collapsing of nodes, edges and elements, the computational mesh requires an extensive topological treatment. Note that due to the separate treatment of the surface mesh and the volumetric nodes, each set of data must be handled in a different way. Additionally, mesh smoothing may be used to reduce the number of operations performed on the meshes.


(a)

(b)

Figure 6.1: Data-set representation of the meshes used in the present numerical code. (a) The surface meshes, which comprise the interface between the fluids and the domain boundary, are passed as an input parameter to the open source library *TETGEN*, which export the 3-dimensional connectivity array. (b) The volumetric mesh is than used to discretize the two-phase flow equations.

6.2 Volumetric nodes

For each time step, the node distribution of the volumetric mesh (tetrahedrons) is monitored and compared to the previous iteration. The tetrahedron edge length determines whether insertion or deletion is required for a given predefined distance in a specific zone, therewith it is possible to avoid clustering and dispersion of computational nodes. In this work, the distribution edge lengths h are obtained by the solution of the following Helmholtz's equation:

$$\nabla^2 h = \frac{1}{k}(h_b - h) \tag{6.1}$$

where k is a diffusive parameter and h_b is the initial edge length distribution. Thus, the obtained solution corresponds to a smooth distribution of nodes in the space. Note that for large values of k in the above equation, the right hand term tends to zero, thus resulting in Laplace's equation ($\nabla^2 h = 0$) in which the solution damps all the sudden changes in the distance between nodes. On the other hand, assuming a small value of k, the solution h approaches the initial point distribution h_b.

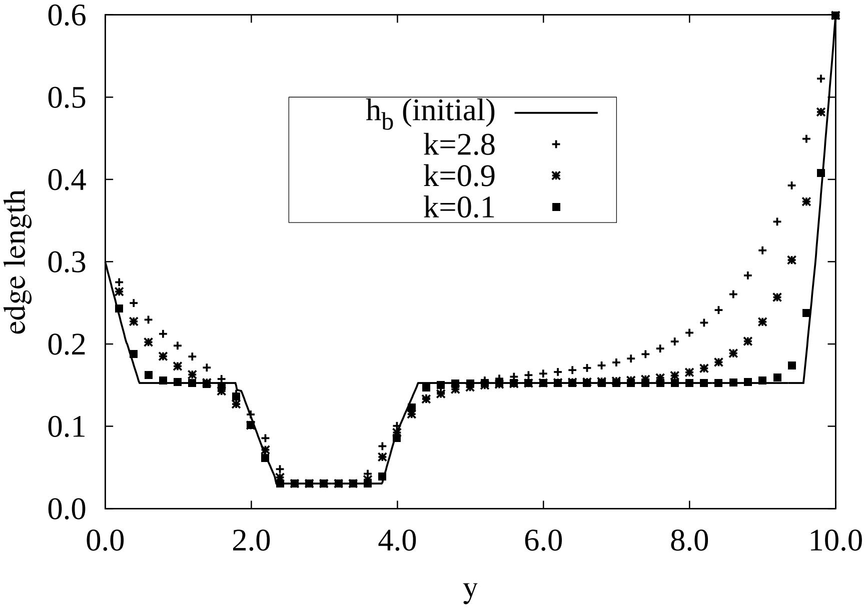
As mentioned before, the initial volumetric node distribution h_b may be set according to the flow requirements, where a particular zone may or may be not refined. Figure (6.2) shows two examples of the solution of the Helmholtz equation. In both cases, the solid lines represent the initial edge length distribution h_b, while the other lines show the solutions for different diffusive parameter k. As can be seen for large values of k, the smoothing effect is more pronounced, while for small values of k, the solution approaches the initial distribution h_b. In Fig. (6.2a), the edge length distribution is seen in the z axis, in which the bubble is located within the interval $z = \{2, 4\}$. This shows a typical distribution of edge lengths in rising bubble simulations where a dense cluster of nodes is expected near the bubble's interface. However, far from the interface, the region is not refined as can be seen with large values of edge length h. In Fig. (6.2b) the edge length distribution differs from the previous case. The aim is to achieve an efficient distribution along a pipe or channel, thus more nodes are required close to the domain boundaries and a coarse mesh is sufficient in the middle. Such a refinement is important to investigate liquid film thickness and bubble/boundary interactions present in annular and bubbly flows.

The solution of the Helmholtz equation (6.1) has been shown to be extremely important in order to achieve a smooth distribution of nodes in the 2-dimensional and 3-dimensional domains. It is discretized through the Finite Element method by following the same procedures described earlier in Chapter 4. However, its continuous solution may overload the computational resources and it may not necessarily bring significant changes in the simulation at each time step. Due to minor modifications in the mesh distribution from one time step to another, mainly driven by small time step, the solution of the Helmholtz equation may be stored in the memory and kept for a few iterations. After a certain number of iterations, say 5, the procedure is redone. Thus, the successive solution of the Helmholtz equation is avoided but a smooth edge length distribution is achieved during all the simulation.

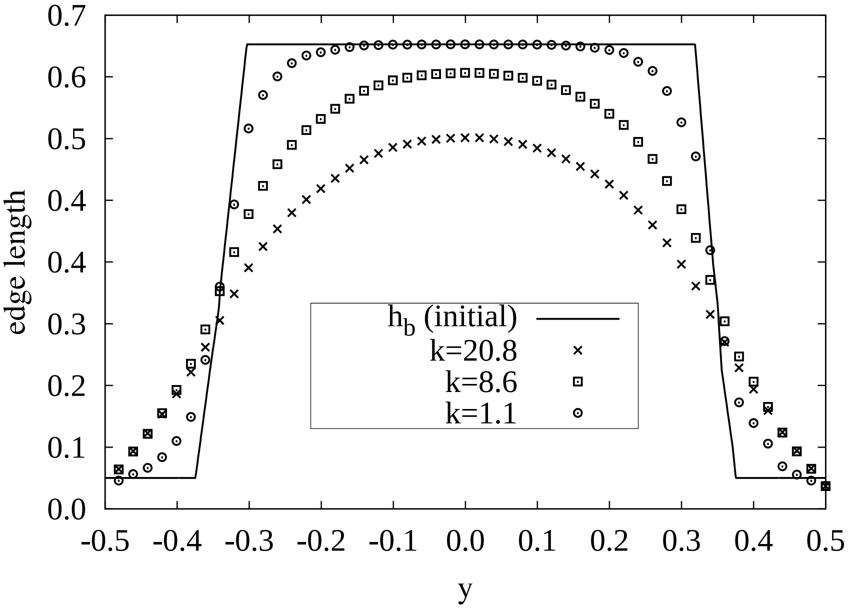
6.3 Mesh smoothing

A proper choice of the mesh velocity and the mesh strategy is necessary to avoid the fast degradation of the computational elements and the onset of non-desirable numerical instabilities found in the pure *Lagrangian* framework. Especially close to the boundaries, the elements are often stretched and compressed in such a way that the simulation becomes unstable. Therefore it is desirable to chose the mesh velocity to rearrange the nodes, thus keeping the elements bounded within good aspect ratios. Additionally, insertion and deletion of vertices should be performed whenever the edges of the elements are greater or smaller than a predefined size. Unfortunately, the re-meshing process and the rearrangement of mesh elements may become very expensive in terms of processing time and therefore they require special attention. To address this issue, a new fast node repositioning scheme is proposed and described here as weel as a new local re-meshing technique that bounds the element aspect ratios within a satisfactory level with no significant time cost in two-phase flows.

As seen earlier in Section (3), the mesh velocity $\hat{\mathbf{v}}$ determines the motion of the nodes of the finite element mesh. This velocity is obtained by a linear combination of two others: the



(a)



(b)

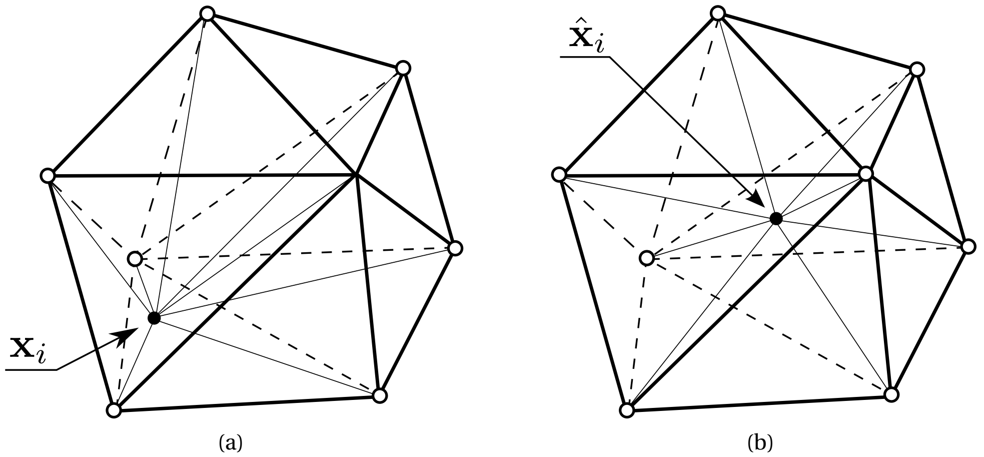
Figure 6.2: Solutions of the Helmholtz's equations for different diffusive parameter k. (a) The sample was taken along the z axis, in which the bubble's location can be seen within the interval $z = \{2, 4\}$. (b) The y component represents the channel cross section. In this case, the mesh is more refined close to the channel's boundaries ($y = -0.5$ and $y = 0.5$) and coarser in the middle.

flow velocity itself and an *elastic* velocity, in which the latter is defined according to some smoothing criteria intended to redistribute the nodes optimally, thus minimizing the number of re-meshing steps and avoiding heavy computation requirements. The transfinite mapping method (see [36]) and Laplacian smoothing are examples of mesh-update procedures, the latter being chosen for this work.

The idea behind the Laplacian smoothing operator is to move the non-uniform mesh nodes by redistributing equally the distance between them, thus achieving a smoothed distribution. In this work, the implementation consists in defining a function in terms of the 1-ring neighboring coordinates; thus the mesh nodes are repositioned in such a manner that the elements satisfy some predetermined geometric criterion. The smoothing procedure is part of an iterative scheme which converges to a more uniform point distribution. According to [22], the new point's position $\hat{\mathbf{x}}_i$ can be approximated using a weighted sum of the 1-ring neighbors of a node as follows:

$$\hat{\mathbf{x}}_i = \sum_{i \in N_1(j)} w_{ij}(\mathbf{x}_j - \mathbf{x}_i) \tag{6.2}$$

where w_{ij} is the weight that can be set as uniform or proportional to the inverse distance from its neighbor vertices and N_1 is the set of 1-ring neighbors of the $j^{th.}$ node. Thus, the mesh velocity $\hat{\mathbf{v}}_e$ is found by dividing the displacement of each node's position $\mathbf{x}_i$ by the simulation time step dt. This approximation is not sufficient to distribute the mesh nodes optimally in one single application step but once applied systematically, the mesh elements converge to a satisfactory shape. Figure (6.3) shows a 3-dimensional example of a node reposition scheme based on the Laplacian smooth operator. As can be seen, the Laplacian operator moves the point in the direction of the polyhedron's centroid, thus the connected edges approach an uniform spatial distribution.



(a)

(b)

Figure 6.3: Laplacian smoothing operation in 3-dimensional space. (a) initial point position and (b) final point position after successively smoothing steps.

So far we have seen that the uniform weighted approximation results in significant sliding and shape distortion in an unstructured 3-dimensional mesh as has been reported by Taubin [88]. Therefore, a scale-dependent Laplacian approximation seems to be ideal for all the 3-dimensional simulations. Its difference, compared to the uniform weighted method, is that $w_{ij} = 1/|e_{ij}|$ where e_{ij} is the distance between the node and each neighbor. Thus, the node sliding is less pronounced and the shape of the elements converges to a more equidistant distribution of vertices. Thus, the final Laplacian smooth distribution is obtained by:

$$\hat{\mathbf{v}}_{e_i} = \frac{\sum_{i \in N_1(j)} e_{ij}^{-1} (\mathbf{x}_j - \mathbf{x}_i)}{dt} \tag{6.3}$$

Additionally, another technique has been tested here which defines the velocity $\hat{\mathbf{v}}$ as a function of the neighbor's velocities instead. This approach is applied to all the volumetric mesh nodes, but only the surface mesh node (interface and boundary) velocities are taken into account. If an iteration process is used, the velocity is spread smoothly over the vicinity of the surface mesh. The advantage of such an approach is that for high velocity gradients, when the elements tend to collapse, the surface mesh velocity is distributed around the neighborhood of each node next to the surface, thus moving them all in the direction of the closest surface node normal vector. The procedure is represented by the following scheme:

$$\hat{\mathbf{v}}_{v_i} = \frac{1}{n} \sum_{j \in N_1(j)} \mathbf{v}_j \tag{6.4}$$

where N_1 is the set of the 1-ring neighbors to the jth node, v_i is the velocity associated to the ith node and n is the number of mesh neighbors to the ith node. The mesh velocity is represented by $\hat{\mathbf{v}}_{v_i}$ is the mesh velocity. Note that with such an approach, the nodes close to the surface mesh (interface and domain boundaries) will behave like those of the surface mesh. Thus, it is expected that these nodes will not collapse nor cross a triangle interface. This scheme has been successfully applied to overcome the fast element distortion close to the interface where the velocity gradient may be high.

Figure (6.4) illustrates the above velocity repositioning scheme in 2-dimensional space and its simplicity. The surface's velocity is scattered gradually to the nodes near the surface, the closer is the node to the surface, the stronger is the influence of the moving interface. This scheme is an important tool to avoid that nodes and surface elements collapse. Note that its extension to 3-dimensional space is straightforward by considering an embedded surface in $\mathbb{R}^3$ as the interface between the fluids.

Moreover, the velocity smoothing scheme proposed here is useful for simulations involving bubble/bubble and bubble/wall interactions. Close to the boundaries, the volumetric nodes

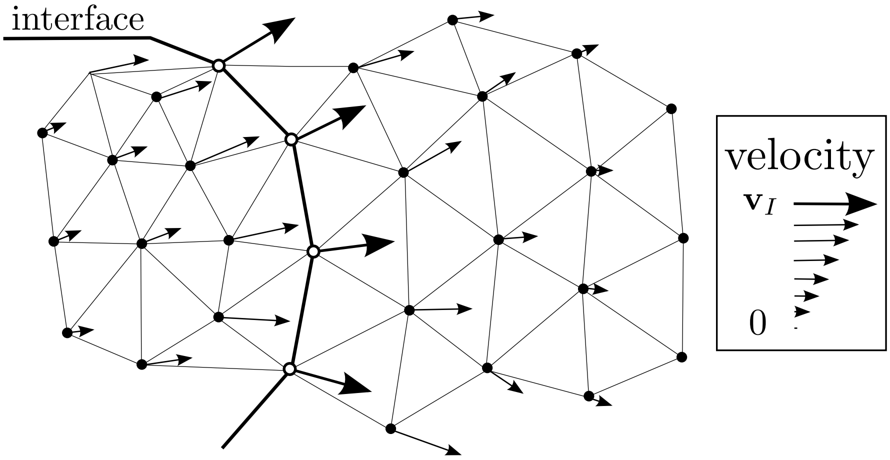


Figure 6.4: Velocity smoothing operation in 2-dimensional spaces. Near the interface, the nodes are more influenced by the surface velocity (large arrows), while if the node is located far from the surface, the mesh velocity $\hat{\mathbf{u}}_v$ is less pronounced (small arrows). Its analogy to 3-dimension space is straightforward by considering a surface embedded in $\mathbb{R}^3$ as the interface between the fluids.

are moved according to the interface's velocity, which is responsible to pull the nodes away, and the zero boundary mesh velocity, which stops the motion of an approaching node. Thus, the volumetric nodes are compressed and squeezed in the gap between the interface and boundary meshes, such that the precision and the number of nodes remain unchanged.

Due to the separation of the domain and the surface mesh in the above procedure, the mesh distribution treatments may be combined into a scheme and adjusted by parameters varying from 0 to 1, which is a 3-dimensional generalization of the approach presented by Souza and Mangiavacchi [84] for 2-dimensional simulations. The domain and surface velocities are therefore treated as follows:

$$\hat{\mathbf{v}}(\mathbf{x}) = \begin{cases} \mathbf{v} - \gamma_1(\mathbf{v}\cdot t)\,t + \gamma_2(\mathbf{v}_e \cdot t)\,t & \text{if } \mathbf{x} \text{ belongs to the interface} \\ \beta_1\mathbf{v} + \beta_2\mathbf{v}_v + \beta_3\mathbf{v}_e & \text{if } \mathbf{x} \text{ does not belong to the interface} \end{cases} \tag{6.5}$$

In such a method, due to the description of the interface mesh by computational elements, the surface should move according to the fluid motion. In the above equation, if $\mathbf{x}$ belongs to the interface, we can define its velocity as $\mathbf{v}_I$. Thus, it is convenient to decompose it into two orthogonal components: $\mathbf{v}_{I_n}$ and $\mathbf{v}_{I_t}$ which represent the normal and tangential velocities, respectively. To decrease the displacement of nodes in the tangential direction, one may remove partially, or even totally, its velocity from the total interface's velocity. This can be achieved by either projecting the interface's velocity $\mathbf{v}_{I_n}$ to the normal vector associated to the node or, in a simpler manner, by removing the tangent component from the total surface mesh velocity $\mathbf{v}_{I_t} = \mathbf{v} - (\mathbf{v} \cdot t)t$. Figure (5.4) shows the mentioned decomposition of the interface's velocity $\mathbf{v}_I$ in to two orthogonal vectors. Such a procedure may be included into a scheme so that the intensity of the tangential velocity can be easy modified. Therewith, the parameter γ_1 controls the magnitude of the tangent velocity in the total interface's velocity. Letting $\gamma_1 = 1$, only the normal interface's velocity is taken into account in the surface mesh motion, and therefore the surface nodes are not allowed to move in the tangent direction. Additionally, the parameter γ_2 includes the smoothing scheme in Eq. (6.4) on the surface mesh nodes, thus keeping them all bounded within good aspect ratio. The parameter β_1 controls the Lagrangian motion of the inner and outer volumetric mesh velocity. By setting $\beta_1 = 1$, the flow velocity $\mathbf{v}$ is fully included in the moving mesh velocity $\hat{\mathbf{v}}$ and, consequently, the volumetric nodes move according to the flow field. Otherwise, letting $\beta_1 = 0$, the flow velocity $\mathbf{v}$ is not taken into account on the moving mesh velocity. The parameters β_2 and β_3 control the intensity of the velocity smoothing scheme $\mathbf{v}_v$ and the laplacian smooth scheme $\mathbf{v}_e$ into the moving mesh velocity. Thus, setting both parameters to null, the volumetric mesh smoothing is not performed. Note that the parameters γ and β may vary from 0 to 1 to achieve a desirable node distributions according to the simulation requirements.

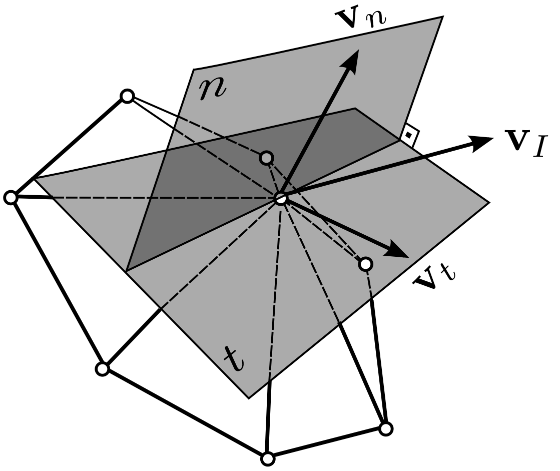


Figure 6.5: Normal and tangent components of the interface's velocity vector. The proposed scheme allows to remove partially or totally the tangent component of the interface's velocity $\mathbf{v}_I$ by varying the parameter γ_1.

The most immediate case which requires very strict mesh control is the simulation of a bubble in a gravity-driven flow. In such a demanding condition, the computed flow field tends to drag the surface nodes from one region to another mainly due to the higher shear stress close to the interface. The flexibility of the parameters presented in Eq. (6.5) allows one to chose $\gamma_1 = 1$, so that the tangential surface velocity is completely eliminated from the equation and therefore the surface nodes are allowed to move only in the normal direction. Additionally, the wake and vortex formed by the bubble's ascension push the surface nodes around the bubble's surface, compressing them one against the other; thus a stiff mesh is recommended to avoid such a problem and this is achieved by setting $\beta_1 = 0$. One may also adjust the parameters β_2 and

γ_2 between 0 and 1 to displace the nodes close to the interface and to optimally redistribute the new surface mesh elements respectively, as well as the value of β_3 to guarantee better tetrahedron aspect ratios.

In contrast to the above example, if the bubble or drop is nearly static or its displacement with respect to the domain is negligible, different mesh parameters are required. Due to the low level of motion of the flow field, the mesh velocity may be set to a pure Lagrangian motion ($\beta_1 = 1$), thus describing the fluid convection with higher precision and a lower number of mesh nodes compared to a standard fixed mesh simulation. The other parameters may be adjusted according to the need to preserve the mesh quality. Moreover, if the flow is controlled by an inflow velocity condition, letting $\beta_1 = 1$ and $\gamma_1 = 0$ may be the appropriate choice for such a problem.

A third case is illustrated for shear-driven simulations. Due to the prescription of a velocity inflow condition, a pure Lagrangian motion is not recommended due to the strong mesh distortions that may appear, especially close to the boundaries. Thus, it is recommended to set the mesh parameter $\beta_1 = 0$. This implies a stiff volumetric mesh. The *elastic* velocity parameters β_3 and γ_2 may be adjusted to 1, thus maintaining the nodes well distributed for both the volumetric and the surface meshes. The parameters β_2 and γ_1 may be arbitrarily chosen to fit the requirements of the mesh.

6.4 Surface remeshing

Unfortunately, mesh smoothing itself is not able to keep all the elements bounded to optimal shapes after numerous iterations. Furthermore, the moving front creates a poor distribution of surface nodes which can affect the accuracy of the computed curvature and, consequently, the final solution. Since the connectivity of the surface mesh is handled by the code, a re-meshing technique is thus required to keep the surface elements aspect ratios in a satisfactory range as indicated by ([34], [95], [41], [46], [89]). The technique proposed here consists of changing the connectivity of surface nodes and elements through “flipping” operations. Additionally, insertion and deletion of nodes is required when a coarse surface mesh is detected or when a dense cluster of surface nodes is not desired, respectively. The new methodology proposed for insertion, deletion, contraction and flipping of triangular edges on the surface mesh is described below.

6.4.1 Point insertion

The strategy for insertion of points aims to occupy “barren” areas and to increase the accuracy in certain regions of the surface mesh where a higher precision is required. Different techniques and insertion criteria can be found in the literature, especially in the computer graphics area, thus it is desired to chose a suitable approach to the investigated problem. In two-phase flows, the flow field formed by the motion of a single bubble tends to move the surface nodes from one region of the interface to another, producing non-uniform node distributions. Furthermore, when the bubble interface moves away from another surface, the 3-dimensional elements are slightly stretched and consequently their aspect ratios change and thus the insertion of nodes may be required to maintain the desired mesh quality.

Figure (6.6) shows two connected triangles with vertices numbered from 1 to 4. When, due to stretching, edge $1-2$ becomes longer than a predefined length, a new node v is inserted in the middle of the edge $1-2$, thus dividing the segment into two equal parts, and consequently creating two new elements which are both part of the surface mesh.

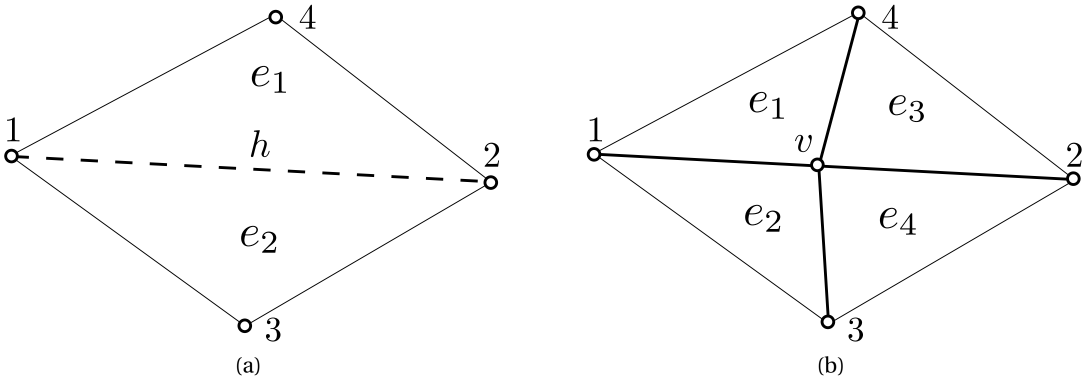
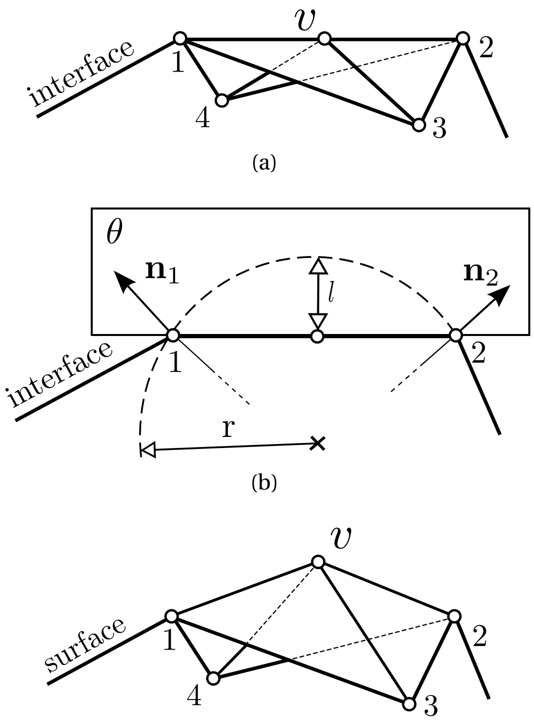


Figure 6.6: Insertion of a surface node. (a) The edge $1-2$, which is longer than a fixed parameter h_{max}, is identified. (b) The new node is then added at the midpoint of the edge $1-2$.

Secondly, if the new node v is inserted on the segment that defines the edge $1-2$ (see Fig. 6.7a), it will introduce a local curvature error that is proportional to l/h, where h represents the triangle edge length and l the distance from the edge $1-2$ and its correct position considering the local mean curvature, thus adding a perturbation that affects the accuracy of the computation of the surface tension force. To minimize such an undesirable error, the new node must be placed according to the curvature of its neighbors. This can be achieved by fitting a circular segment which passes through the vertices 1 and 2. As exemplified in Fig. (6.7b), the θ-plane may be defined by the mean normal vector of two adjacent triangular elements, namely $1-2-3$ and $1-4-2$. The normal vector associated to the nodes 1 and 2 are projected onto the θ-plane forming $\mathbf{n}_1$ and $\mathbf{n}_2$ respectively. The intersection the line of action of these two vectors is chosen as the approximate center (x_c, y_c) of a circumference $(x-x_c)^2+(y-y_c)^2=r^2$ with radius r. Thus, the solution of this equation is used to determine the displacement l, and consequently the final position of the new node v, as can be seen in Fig. (6.7c).

6.4.2 Point deletion

The displacement of vertices in the moving mesh technique may cause the dense clustering of nodes in particular areas of the domain, including the triangular surface mesh. Consequently,



(a)

(b)

(c)

Figure 6.7: Representation of the 3-dimensional triangular surface mesh. (a) The node v is added at the midpoint of the edge $1-2$ (b) The plane θ is derived by the mean of two element normal vectors which are adjacent to the edge $1-2$. The vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ are the projection of the normal vectors of nodes 1 and 2 onto the plane θ. (c) The node's new position is found by moving it from the edge $1-2$ toward the circle segment in (b), thus the curvature error in v is reduced.

with the regrouping of these nodes, the elements become smaller and the time step size decreases due to the *Lagrangian* motion restriction. Additionally, the total number of nodes belonging to the mesh increases, increasing the processing time. The additional number of vertices does not necessarily bring real benefits to the accuracy of the final solution and thus these unnecessary vertices should be eliminated. There are many ways to delete a single mesh point and conserve the mesh quality for the next iteration. The present work has adopted two different techniques to keep the mesh bounded to the Delaunay properties. Both approaches attempt to find an edge h that is smaller than a predefined length h_{min}. Once such an edge is detected, the method computes the sum of the edge length of the 1-ring neighbors of both extremity vertices. The one which has the lowest value is then considered for elimination from the surface mesh. Once the node deletion is performed, its neighbors form a polyhedron which must be subdivided to recover the triangular structure. Figure (6.8) shows the deletion of a surface node and the polyhedron that should be remeshed.

Two strategies were tested to reconnect the empty space left by the deletion of the surface

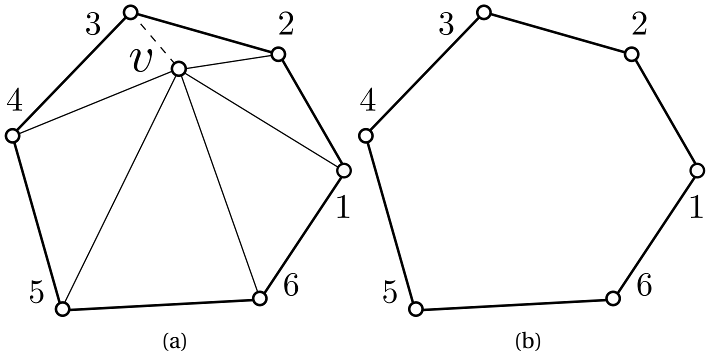


Figure 6.8: Deletion of a surface node. (a) The edge $3 - v$ is detected when its length is smaller than a reference length h_{min}. Due to the sum of neighbor edge lengths, the node v is chosen to be deleted. (b) Therefore, the empty polyhedron must be reconnected to achieve a new surface triangulation.

node v. The first strategy is simple and its implementation is straightforward. Let us consider the polyhedron $P = \{1,2,3,4,5,6\}$ as show in Fig. (6.9). The node 1 is chosen to reconnect successively the nodes 3, 4 and 5 by creating edges on the surface mesh. If the polyhedron nodes have some orientation defined, each triangle may be created by choosing two successive nodes. Thus, the first triangle is formed by connecting the nodes $\{1,2,3\}$ and the second by $\{1,3,4\}$. The strategy continues to set the third triangle $\{1,3,5\}$ and the last one is than defined by $\{1,5,6\}$.

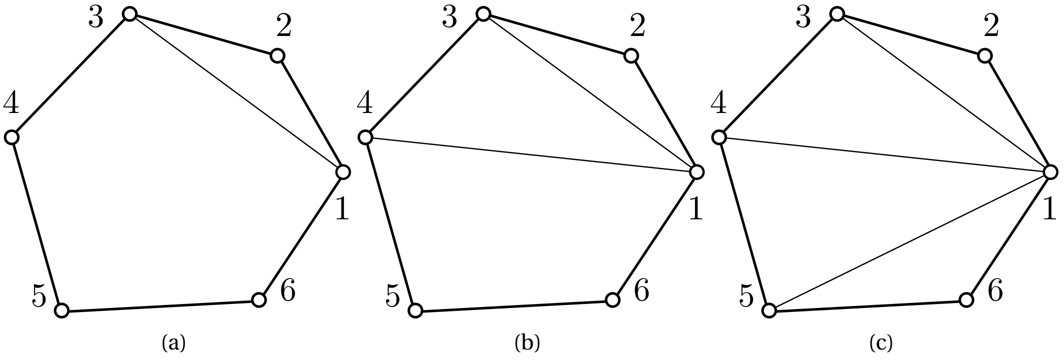


Figure 6.9: Remeshing of a surface polyhedron by successive node re-connections. (a) An edge is created by connecting the nodes 1 and 2. (b and c) The node 1 is then connected to the remaining nodes 4 and 5, thus achieving the final surface triangulation.

The second strategy is done based on the work of Devillers [23] and Xu et al. [100] and extended here to triangular surface meshes. Let us consider a generic polyhedron $P = \{x_0, x_1, ..., x_k, x_0\}$, where the first "ear" of the polyhedron P is defined by the triangle with vertices $x_i - x_{i+1} - x_{i+2}$. Such an ear will be part of the surface triangulation if and only if the segment $[x_i, x_{i+2}]$ is located inside the polyhedron and it does not intercept its boundary. A sub-set of P is formed with the deletion of the point x_{i+1}, and then the new triangular “ear” may be found by repeating the described strategy. The process is iterated until the number of nodes of the sub-set of P is equal to 3, therefore the last triangular ear is composed by $x_{k-1} - x_k - x_0$.

Figure (6.10) shows the schematic representation of the deletion process and remeshing by the “ear” technique. The edge $3 - v$ is detected as being smaller than h_{min} and the node v is chosen to be removed. The neighbor triangles of node v are eliminated from the surface mesh and the polyhedron formed by the 1-ring neighbor nodes of v is used to remesh the empty space. In this example, the reconnection of nodes to form the new triangulation is done by defining a polyhedron $P = \{1,2,3,4,5,6\}$ and the first “ear” to be $E_1 = \{1,2,3\}$. The first triangle is created and the node 2 is deleted from P. The new sub-set of P is defined as $P_{s1} = \{1,3,4,5,6\}$ and thus the next “ear” as $E_2 = \{3,4,5\}$. Node 4 is then deleted from the sub-set P_{s1}, thus creating $P_{s2} = \{1,3,5,6\}$. A new ear is set on the triangle $E_3 = \{5,6,1\}$, the node 6 is deleted from the sub-set P_{s2} and consequently the new sub-set $P_{s3} = \{1,3,5\}$ is assembled. Since the number of nodes in $P_{s3} = 3$, the last triangle is formed and the local re-meshing is accomplished.

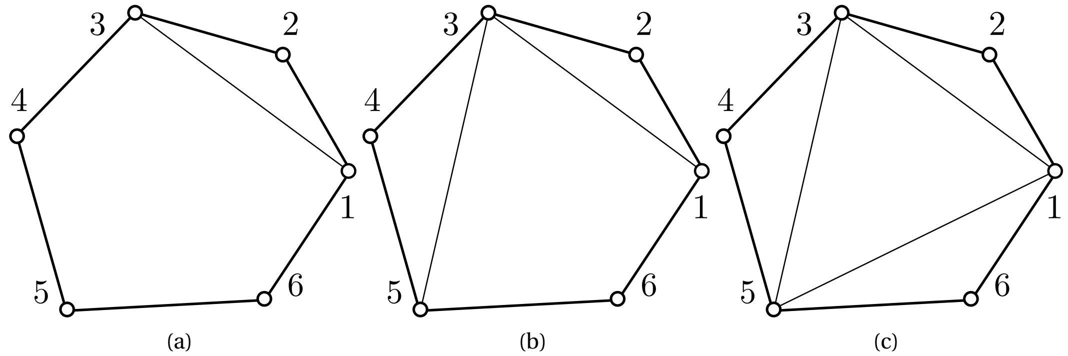


Figure 6.10: Reconstruction of the surface mesh by the “ear” technique. (a) First “ear” is achieved by connecting the nodes $1-2-3$ and forming the surface triangle. The node 2 is then deleted from the polyhedron P. (b) The new triangle is formed by connecting the nodes $3-4-5$, and thus the node 4 is eliminated. (c) Last two triangles are created from nodes $5-6-1$ (node 6 is deleted) and $1-3-5$, which are the remaining nodes of the successively deleted polyhedron.

The strategies presented above are not sufficient to guarantee optimal triangular shapes. In fact, the selection of the initial node should be treated with care by considering the polyhedron spatial geometry. For instance, if the polyhedron has a concave shape and the initial node is wrongly chosen, the remeshing procedure will invalidate the triangulation by creating non-triangular elements or even creating elements outside the polyhedron boundaries. Moreover, flipping operations may be required to achieve an optimal triangle distribution. Such an operation was implemented in this work and it will be described in the next sections.

6.4.3 Edge contraction

This strategy is based on the contraction of an edge and it is well discussed by [62]. Once an edge h for each $h < h_{min}$ is detected, this scheme aims to collapse the two extremity vertices into the midpoint of the edge h; thus the two adjacent triangles are removed from the surface mesh, as shown in Fig. (6.11). After it has been contracted, the edge h is no longer part of the surface mesh, so that nodes 1 and 2 occupy the same position while nodes 3 and 4 remain at their locations. The benefit of such an approach, compared to the previous point deletion strategy, is its geometrical simplicity since the surrounding connectivity of the mesh is not affected.

As can be seen in Fig. (6.12), the same consideration of the curvature of two adjacent nodes found in the insertion strategy should be taken into account when collapsing two vertices, thus avoiding displacement errors and undesirable loss of mass. This is done by fitting the equation of a circle to the nodes 1 and 2 and considering the curvature of the adjacent nodes $(1-2-3-4)$. Thus, the collapsed node is displaced according to the neighbor curvature values.

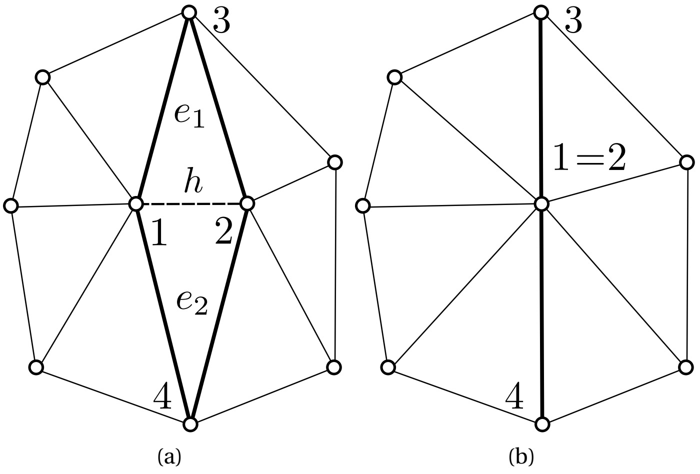
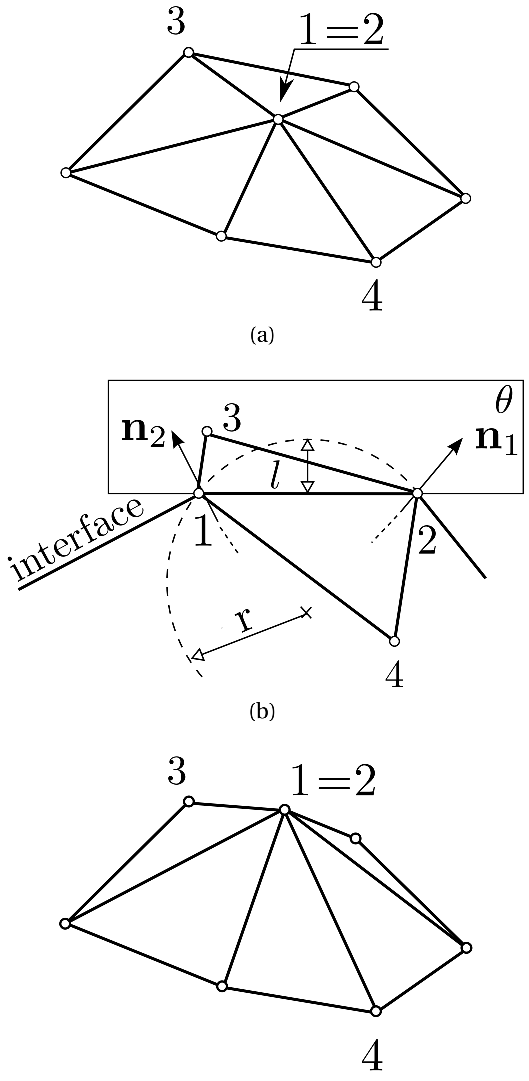


Figure 6.11: Contraction of a surface edge. (a) The edge h (segment 1-2) is found to be smaller than h_{min} and (b) so it is collapsed to the midpoint of the same edge. Due to its simplicity, only triangles e_1 and e_2 are eliminated from the surface mesh and the remaining node connectivity is not affected. The new location of node 1 should respect the curvature of its neighbors as described in the insertion strategy.

6.4.4 Edge flipping

Since insertion, deletion and collapsing of vertices may deteriorate the mesh, edge flipping may be required to restore the mesh quality. Such an operation in a 3-dimensional surface is more restrictive compared to 2-dimensional spaces. The flipping criteria implemented here considers four different measurements:



(a)

(b)

(c)

Figure 6.12: Node displacement according to neighbor's curvature in the process of edge contraction. (a) The plane Ω is found using the curvature vectors of nodes 1 and 2, thus a circle equation is fitted and (b) its solution is used to displace the node and avoid losses of mass. (c) The resulting scheme of edge contraction considering the neighbor's curvature.

- sum of the triangle aspect ratios;
- curvature of neighboring nodes;
- sum of the triangle areas;
- circumcenter of each triangle.

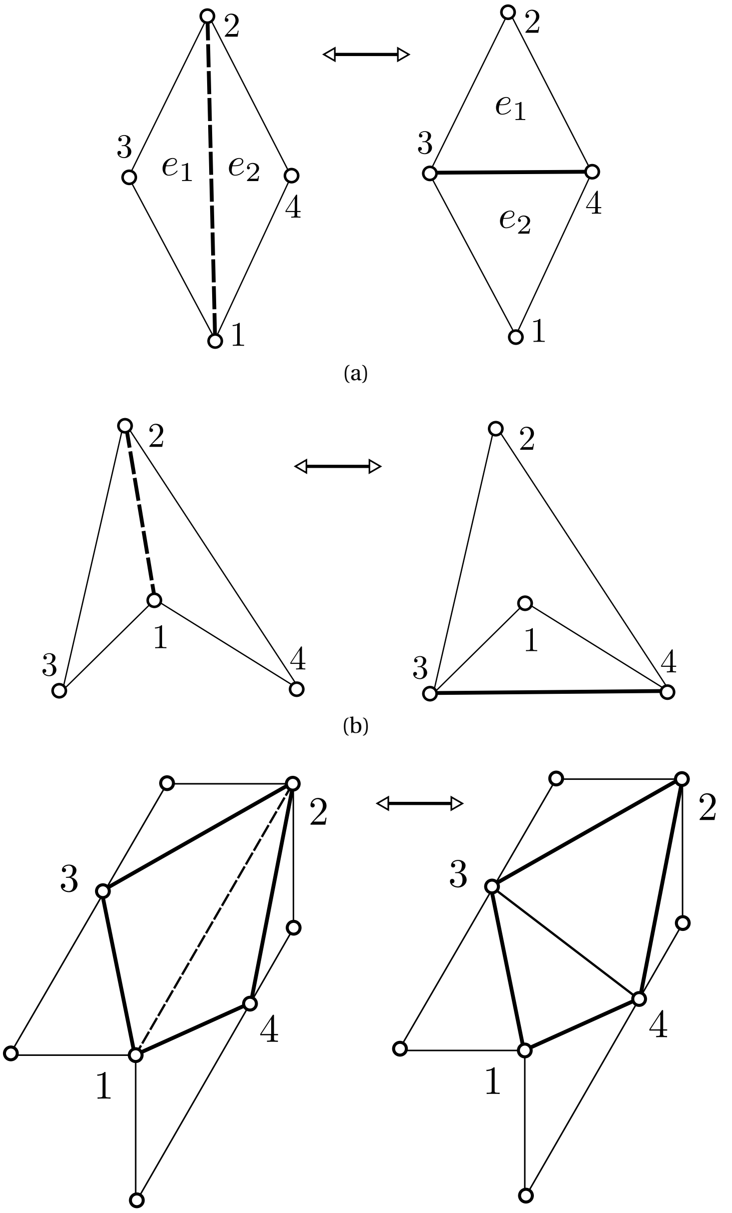
These parameters are evaluated at every time step to check if the flipping operation will be performed. This is achieved by comparing the quality of the initial and the modified pair of triangles, thus if one criteria fails the flipping is not performed.

In the literature, there are may ways to check the triangle aspect ratio such as edge ratio measurement, relative size squared, maximum and minimum angle, etc. (see [81], [28] and [23]). The one chosen in this work considers the radius of the inscribed circle and the longest edge length. This scheme provides quantitatively the quality of a given triangle and thus can be used as parameter to the flipping operation. The neighbor's curvature should be also considered before flipping an edge due to a strong restriction on the surface embedded in $\mathbb{R}^3$. If the curvature is too high, the flipping operation can damage the surface mesh, thus forcing the simulation to shut down. In this work the curvature limiter has been adopted to be $\kappa_{max} = 40$, i.e. above this limit flipping is not performed. Additionally, the sum of the triangle areas are taken into account. This measure restricts the flipping operation if the resulting sum of the areas is smaller than before flipping. Finally, the circumcenter of each triangle is also taken into account as a quality ratio parameter to the final local mesh. Note that these flipping parameters are required to avoid strong mesh degradation and large losses of mass. However, the flipping operation is especially required to keep the mesh bounded to within a satisfactory aspect ratio.

Figure (6.13a) shows a typical flipping operation done on the surface edge. According to the criteria mentioned above, the triangles with vertices $1-2-3$ and $1-4-2$ have lower ratio quality compared to the triangles with vertices $1-4-3$ and $3-4-2$, thus the flipping operation is performed and the new mesh is achieved. On the other hand, Fig. (6.13b) shows a case where a diagonal flip should be avoided since it contradicts the triangular surface mesh. The new generated element $3-4-1-2$ is not triangular and the surface mesh is consequently corrupted. As pointed out by [76], pronounced loss of mass occurs if the flipping operation is performed when the angle of two consecutive faces is lower than $90°$, where the resulting elements may have a better aspect ratio but the loss of mass may reduce significantly the precision of the simulation (see Fig (6.13c)).

6.4.5 Volume Conservation

Due to the constant surface mesh treatment, the front-tracking methods are known to accumulate spurious loss of mass during a simulation. However, for incompressible flows, the volume of both phases should remain constant and excessive geometrical operations may lead to loss of mass. To avoid such an issue, one should minimize the number of flipping operations and limit the deletion and insertion of new points. In certain cases where the surface mesh is at constant shear, the number of geometrical operations cannot be reduced. As mentioned before, successive mesh smoothing and the displacement of a new inserted node according to its curvature should be performed. Nevertheless, due to the inherent truncation and rounding errors, the combined mass of the two phases may slightly change. Therefore, a



(a)

(b)

(c)

Figure 6.13: Triangular surface flipping operations: (a) The triangle aspect ratio, the curvature of neighboring nodes, and the triangle circumcenter are taken into consideration to perform the flipping from edge $1-2$ to $3-4$; (b) the flipping of edge $1-2$ cannot be assigned due to an inconsistent mesh generation. (c) The flipping operation may lead to local loss of mass and it should be treated with care.

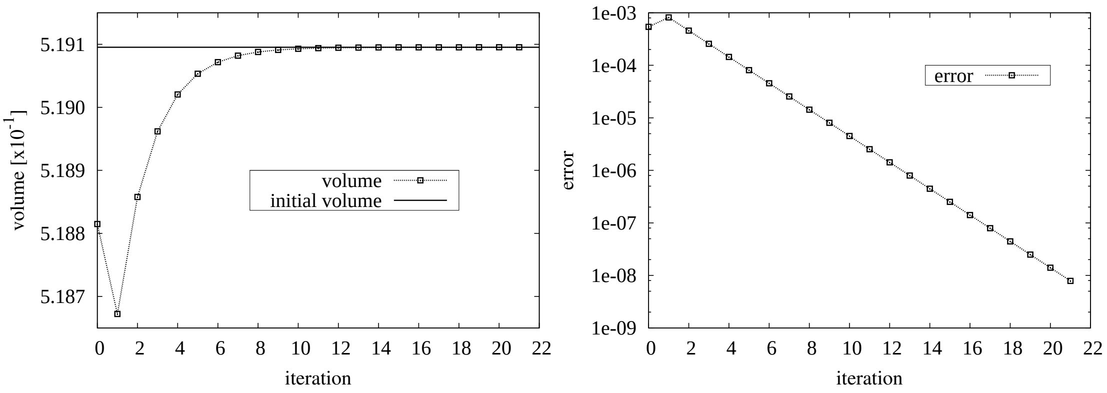
simple treatment has been implemented to compensate for the spurious mass variation, thus avoiding the accumulation of mass conservation errors. This correction is done by moving the surface nodes in the direction of their normal vector. Such a displacement is calculated based on the initial phase volume, which is compared to the current iteration; thus a successive relaxation method is applied to find the final node's positions. The difference between the initial phase volume and the current time step volume may be chosen according to a given tolerance, in which, for the present work, is on the order of 10^{-8}. Note that this tolerance has been chosen according to the simulation experience acquired on the test cases studied in this thesis, however this tolerance may be changed if more precision is required. The volume conservation algorithm is described below.

```
Input: Initial surface volume
TOL = 1.0E-08;
while absolute(error) > TOL do
    foreach surface node i do
        xNormal(i) = normal vector component of node i;
        edge = local surface edge length size;
        x(i) = x(i) + xNormal(i)*edge*error;
    end
    surfaceVolume = compute surface volume;
    error = 1.0 - surfaceVolume/initSurfaceVolume;
end
```

Algorithm 1: Surface volume correction.

Figure (6.14) shows the result, in one time step, of the mass conservation for the rising bubble test case. As can be seen, the current bubble's volume is different of the initial bubble's volume in the iteration 0. This is due to geometrical operations on the surface mesh such as edge flipping, deletion and insertion of nodes. In this time step, 21 iterations are required to restore the initial bubble's volume with an error of order of 10^{-8}. Note that the local surface edge length size should be taken into account for simulations of many bubbles with different mesh refinements.

The geometrical operations on the surface and volumetric meshes are performed preferably at all time steps, as well as the volume correction algorithm and consequently the adaptive mesh refinement is successfully achieved. Thus, the surface and the volumetric meshes are corrected, the phase volumes are adjusted, and the simulation can reach the final state with no significant loss of mass.



(a) (b)

Figure 6.14: Volume correction in the rising bubble test case: (a) The initial volume $V \neq 0.5191$ is computed when the simulation starts, then it is compared to the current bubble's volume and corrected after a few iterations. (b) Convergence error of bubble's volume correction.

7 Validations and Results

This chapter describes numerical results for incompressible two-phase flows obtained with the present Arbitrary Lagrangian-Eulerian Finite Element code. The computed curvature κ is compared to the analytical value of some well-known geometric objects and presented first in this chapter. A number of single phase flow benchmark tests were carried out to successfully validate the code, these included a *Poiseuille* flow, a backward facing step, a lid-driven cavity and a rotating disk flow (see [4] and [74]).

Next, static and dynamic two-phase flow test cases were carried out to validate the discretization procedure and its implementation; these will be described in the following sections. These tests are important to evaluate the coupling of the implemented surface tension force model with other terms of the momentum equation. In doing so, the accuracy of the proposed methodology can be estimated, as well as the precision of the computed curvature.

The tests performed were divided in two groups, namely *static* and *dynamic* tests. The former includes the static droplet and the sessile drop tests in which the convection term is negligible. The latter includes the oscillating droplet, the falling drop and the rising bubble. The oscillating, sessile and falling drop results are compared to analytical solutions, and the rising bubble test is compared to experimental results found in the literature. Moreover, the effect of wall on the bubble dynamics is tested by comparing the 3-dimensional numerical simulation to a well-known flow pattern map found in the literature. The mechanisms of collision of two equal-sized drops are also investigated and, finally, results on microscale flows are presented.

7.1 Curvature calculation

The accuracy of the methodology used in this work to calculate the curvature and the surface tension force, explained in detail in Chapter 5, is strongly linked to the number and size of the elements connected to each single surface node. Thus, in order to validate the computation of the curvature, the relative error, for a given shape, was determined as a function of the characteristic element edge length h of its surface. The relative curvature error (Error_κ) was

Table 7.1: Standard deviation (SD) and error of curvature (κ) for different geometries.

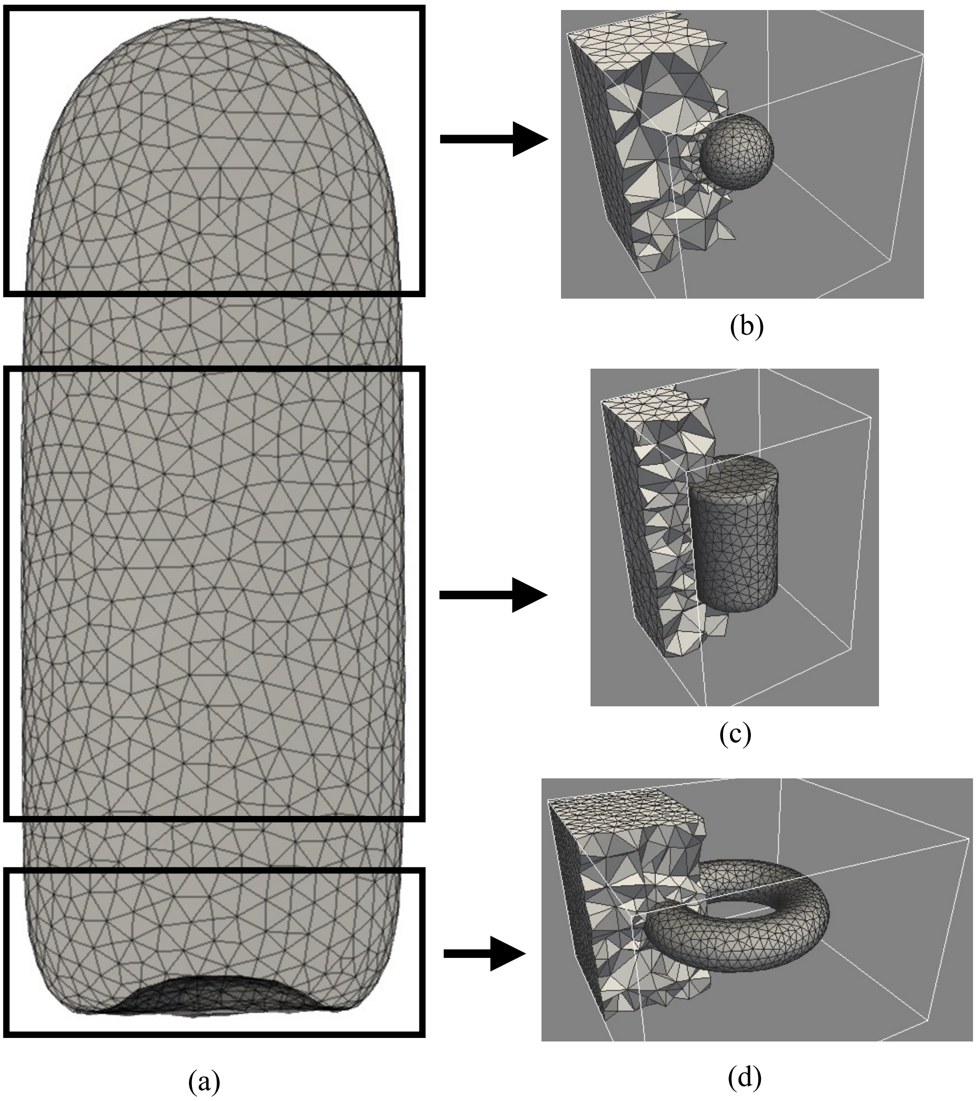
Edge Length	Sphere		Cylinder		Torus	
$[h]$	SD	Error[%]	SD	Error[%]	SD	Error[%]
0.19	0.1487	0.7717	0.2461	3.5625	0.6571	4.5625
0.16	0.1398	0.6304	0.2539	3.7878	0.5645	4.7812
0.12	0.0975	0.3896	0.3515	2.6785	0.5822	3.8554
0.09	0.0715	0.2558	0.2916	2.2232	0.5560	2.3244
0.06	0.0447	0.1231	0.3413	2.3391	0.5588	2.9176
0.03	0.0209	0.0279	0.2979	2.3237	0.5671	3.3712

estimated as follows:

$$\text{Error}_\kappa = \sqrt{\frac{\sum(\kappa_i - \kappa_A)^2}{\sum(|\kappa_i|)^2}} \tag{7.1}$$

where κ is the numerical mean curvature of each surface node and κ_A is its analytical value for a particular shape. The summation is done over all the surface nodes, and thus an average value is computed. The tests were performed and validated against three representative shapes of significance to two-phase flows: a sphere, a cylinder and a torus, due to the value and sign of their curvature. Figure (7.1) shows a comparison of the three mentioned representative shapes against a simulated Taylor bubble. As can be seen, each part of the Taylor bubble assumes different shapes, therefore each case is modeled separately to evaluate the curvature against their analytical values. The representative surface shapes are shown with a portion of the volumetric meshes, intentionally sliced for illustration purposes. The results are shown in Table (7.1) as a function of the average length h, which varies from 0.19 down to 0.03 for each shape. The standard deviation (SD) and the error of the computed curvatures κ are evaluated using their analytical solution, demonstrating that for the cylinder and the torus, where the curvatures change value and sign, the calculation is less accurate if compared to the sphere, but nevertheless they are all bounded to within a maximum relative error of 5% of κ_A.

Additionally, a plot of the computed curvatures of a sphere with a non-dimensional radius $R = 0.5$ is shown in Fig. (7.2). The test was performed varying the maximum admissible surface edge length from 0.03 to 0.30 and the results are compared to the first and second order of convergence curves in a logarithm scale. As can be seen, the curvature calculation for different edge mesh sizes appears to be of first order.


(b)

(c)

(a)

(d)

Figure 7.1: Comparison of a Taylor bubble and three representative shapes used to evaluate the curvature error of the proposed numerical method: (a) Taylor bubble, (b) sphere, (c) cylinder and (d) torus. The sphere with radius $R = 0.5$ has both curvatures with same value and sign. In the cylinder, part of its shape (curved) has one of the principal curvatures zero (along its height) and the other is inversely proportional to the cylinder radius $Rc = 0.5$. The torus has the principal curvatures with opposite signs in its inner part and same positive sign for the remaining shape.

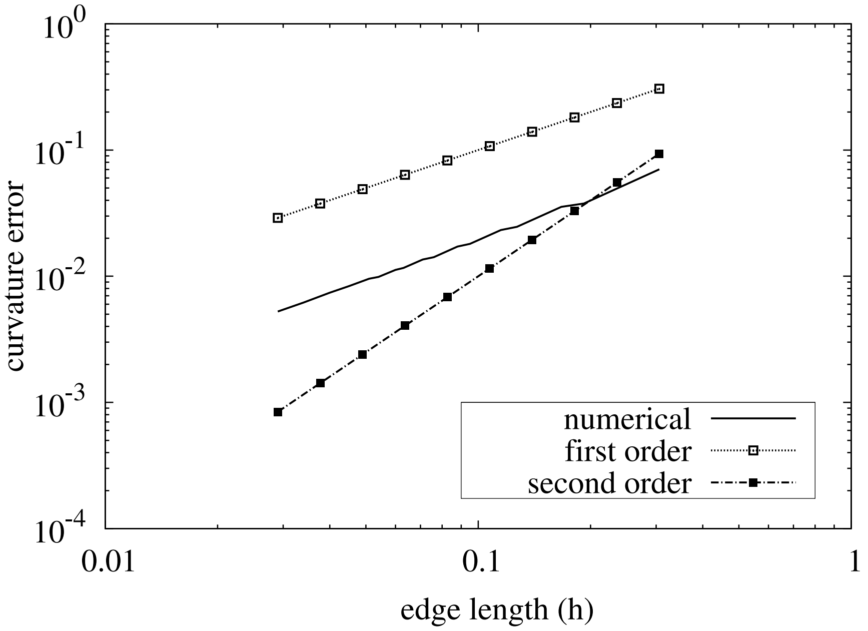


Figure 7.2: Log scale graph showing the convergence order of the new methodology for computing the curvature of a surface. The slope found in the numerical method suggests its convergence is first order.

7.2 Static droplet

The coupling between pressure and surface tension is assessed here with respect to the generation of spurious flow by the simulation of a spherical droplet immersed in another fluid in the absence of any external force and velocity field. From the mathematical point of view, the Navier-Stokes equation reduces to:

$$\nabla p = \mathbf{f} \tag{7.2}$$

This equation shows that the surface tension force is at an equilibrium state with the pressure field. In fact, this is not the equation numerically solved but due to the prescription of zero velocity at the boundaries and the setting of the gravity term to null, all the related terms vanish, and thus they do not influence the final solution. Theoretically, the pressure gradient should completely equilibrate the surface tension force but, due to numerical reasons (coupling between velocity and pressure, accuracy of the numerical scheme, curvature calculation, etc.) spurious currents may appear. It is clear that undesired artificial behaviour, namely spurious currents, should be minimized. To successfully achieve such a difficult task, some authors have suggested different approaches as reported by [85] and [31]. The consistent numerical treatment of the surface tension term as well as a good coupling between pressure and velocity

Table 7.2: Comparison between edge length size, pressure distribution and spurious currents.

Edge Length (h)	N. Surf. Triangles	Δp	Δp_{error}	$\max\{\lvert u\rvert, \lvert v\rvert, \lvert w\rvert\}$
0.10	804	19.98	0.1%	4.1×10^{-3}
0.09	948	19.98	0.1%	1.2×10^{-3}
0.06	2044	19.99	0.05%	8.2×10^{-4}
0.04	2632	20.02	0.1%	3.7×10^{-4}
0.03	4104	20.01	0.05%	1.2×10^{-4}

fields of the present approach are responsible for very weak spurious currents without any supplementary artificial treatment. Thus, simulations have been performed for different mesh refinement levels to quantify such an artificial velocity as well as the error of the pressure field solution inside and outside the droplet, considering the same capillary-viscous length $R_v = 0.002$ as reported by [94], where $R_v = \rho v^2/\sigma$ and v is the kinematic viscosity. In their case, the surface tension coefficient was $\sigma = 5$. Table (7.2) provides a compilation of the results for this test. It can be seen that the error in pressure difference Δp_{error} is approximately linear and the same applies to the spurious currents. Other tests have shown that such values of spurious currents do not considerably affect the final solution of the simulation.

The theoretical value of the pressure difference between the phases is given by the non-dimensional Young-Laplace equation, $\Delta p_a = 2/(\mathbf{Eo}R)$, where R stands for the droplet radius and **Eo** is the *Eötvös* number. The average pressure difference of the numerical result is calculated by summing the values in each phase and dividing by its respective number of nodes as:

$$\Delta p = \frac{\sum_{i=1}^{n_{in}} p_{in}}{n_{in}} - \frac{\sum_{i=1}^{n_{out}} p_{out}}{n_{out}} \tag{7.3}$$

The pressure inside the droplet is given by p_{in} and the number of nodes by n_{in}. The outside phase is describe by p_{out} and n_{out}, respectively.

Figure (7.3) depicts the linear pressure distribution between the phases and the standard *hat* shape inherent to the pressure field for different mesh refinements. These values were taken using a linear interpolation along the z-axis with a unidimensional uniform grid. For all the cases, they were simulated using the same set of parameters but varying the number of the surface mesh elements in a domain $8D$x$8D$x$8D$ where D is the droplet diameter. As expected, the higher is the number of mesh nodes the more accurate is the pressure jump across the interface. This can be seen in Fig. (7.4) where the plot shows the distribution of pressure. Considering the static nature of this test and the velocity recirculation near the drop interface, the mesh parameters used in these simulations were $\beta_1 = 0$, $\beta_2 = 0$, $\beta_3 = 0.1$, $\gamma_1 = 0.0$ and $\gamma_2 = 0.0$, thus keeping the surface and volumetric mesh nearly static.

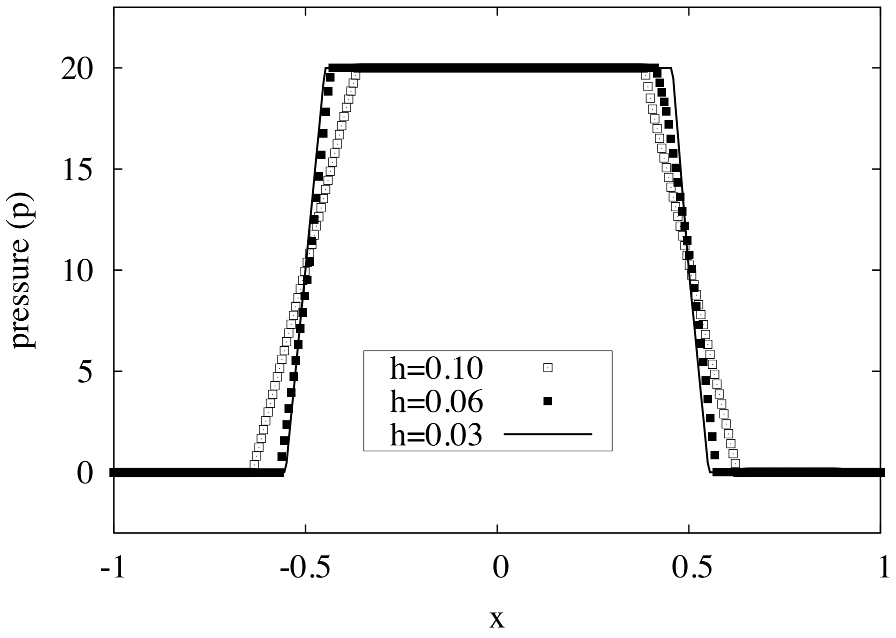


Figure 7.3: Chordal pressure jump between the phases for different surface edge lengths. The solution of the pressure field for a static droplet immersed in a low viscous fluid was interpolated in a linear uniform mesh where the non-dimensional pressure $p = 0$ corresponds to the area occupied by the gas phase and the pressure $p = 20$ stands for the area occupied by the droplet.The test was performed considering the non-dimensional radius $R = 0.5$ and $Eo = 0.2$, resulting $\Delta p = 20$.

7.3 Sessile drop

The next simulation was performed to validate the surface tension implementation and its coupling with pressure and gravity. A spherical drop with radius $R = 0.5$ was initialized two diameters above the bottom of the domain and then released. Due to gravity, the drop, being heavier than the surrounding fluid, falls and hits the solid surface. Before the contact between the solid surface and the interface, the drop deforms to a quasi-steady state and approaches the wall with no significant shape changes. The drop's profile remains axisymmetric and its shape can be approximated by the classical Young-Laplace equation of capillarity which, in

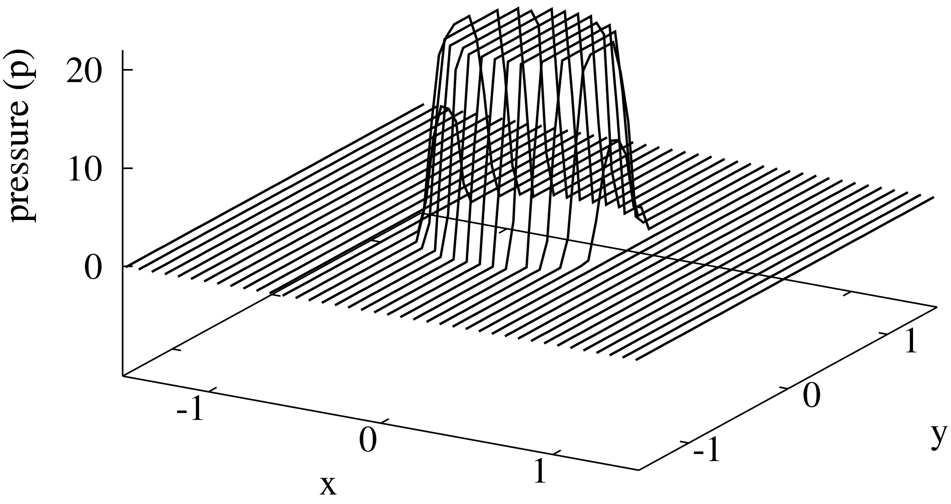


Figure 7.4: Capillary pressure of a spherical droplet immersed in another fluid. The jump in pressure can be seen at the location of the interface.

non-dimensional form, states that:

$$(\frac{1}{R_1} + \frac{1}{R_2})\frac{1}{Eo} = \Delta p_g = \Delta \rho g(z - z_0) \tag{7.4}$$

where R_1 and R_2 are the two principal radii at the apex of the drop, σ is the interfacial tension and Δp_g stands for the hydrostatic pressure difference across the interface, where $\Delta\rho = \rho_{in} - \rho_{out}$. Considering ϕ as the drop tangent angle with respect to the wall, for an axisymmetric drop:

$$(\frac{1}{R_1} + \frac{1}{R_2}) = \kappa = \frac{d\phi}{ds} + \frac{sin\phi}{r} \tag{7.5}$$

where s is the coordinate along the surface and r is the radial coordinate. Substituting Eq. (7.5) into Eq. (7.4), results in:

$$\frac{d\phi}{ds} = \hat{\mathrm{Eo}}(p - z) - \frac{sin\phi}{r} \tag{7.6}$$

$$\frac{dr}{ds} = cos\phi \tag{7.7}$$

$$\frac{dz}{ds} = sin\phi \tag{7.8}$$

Here, $\hat{p}$ stands for the dimensional reference pressure, $p = \hat{p}/\rho g L$ as the non-dimensional pressure and $\hat{\text{Eo}} = \text{Eo}(\Delta\rho/\rho_{in})$ as the modified *Eötvös* number. These equations are solved by the Runge-Kutta method with appropriate initial conditions: $s = 0$, $\phi = 0$, $r = 0.258$ and $z = 0$ and are then solved and integrated up to $\phi = \pi$.

The simulation of the drop was performed assuming the following non-dimensional parameters: $R = 0.5$, Eo = 2, $\rho_{in} = 1.0$ and $\mu_{in} = 1.0$ for the drop and $\rho_{out} = 0.1$ and $\mu_{out} = 0.9$ for the external fluid. The domain limits were set to be $8D$x$8D$x$4D$, where the last dimension stands for the direction of gravity, and discretized by approximately 26000 tetrahedrons and 5100 nodes. The interface mesh had approximately 5400 triangles in which 2056 were part of the interface mesh. The mesh parameters used were $\beta_1 = 0$, $\beta_2 = 1.0$, $\beta_3 = 1.0$, $\gamma_1 = 0.1$ and $\gamma_2 = 0.0$, thus keeping the volumetric and surface meshes as well as the distance of nodes to the surface elements almost uniform while the drop is moving downward to the bottom of the domain.

Figure (7.5) shows the curvature distribution along the drop's height. The solid line was fit using the data from $z = [0.2, 0.8]$ by the least squares method; therefore the slope of such a line gives the value of $\hat{\text{Eo}} = 2$ for which it can be used to solve the previous mentioned ODE. Additionally, comparing the distance from the numerical data to the solid line, it is possible to compute the error of curvature for the simulated drop. Thus, the curvature error Error_κ was found to be 0.2, showing reasonable agreement to the analytical solution. Furthermore, the numerical solution of the drop's shape is compared to the semi-analytical solution of the Young-Laplace's equation (Eq. (7.6),(7.7) and (7.8)) and is shown in Fig. (7.6). The results show that the sessile drop is correctly predicted by the present implementation, due to the accurate balance of the gravity, pressure and surface tension forces.

7.4 Oscillating drop

The evolution of a single drop initially perturbed in its shape is presented below. This case is part of the set of standard benchmarks tests required to evaluate the modeling of the surface tension force. The simulation consists of an initially axisymmetric ellipsoidal drop immersed in another fluid in the absence of gravity. Due to the absence of external forces and assuming the viscous effects to be small, the oscillating process is mainly driven by interfacial forces which are directly balanced by the convection term. The perturbed drop tends to oscillate

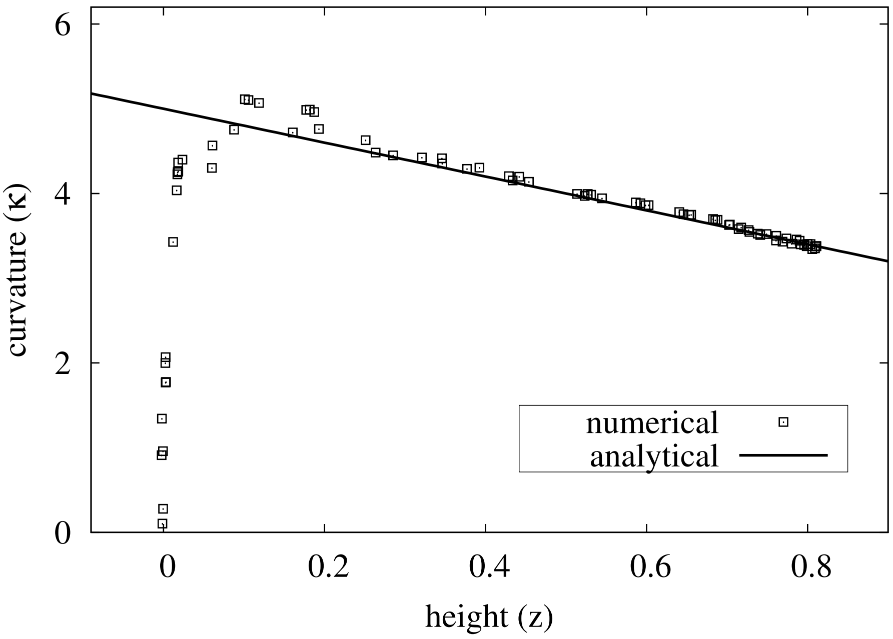


Figure 7.5: Curvature distribution along the drop's height. The solid line was fit by the least square method and its slope gives the value of $\hat{Eo} = 2$.

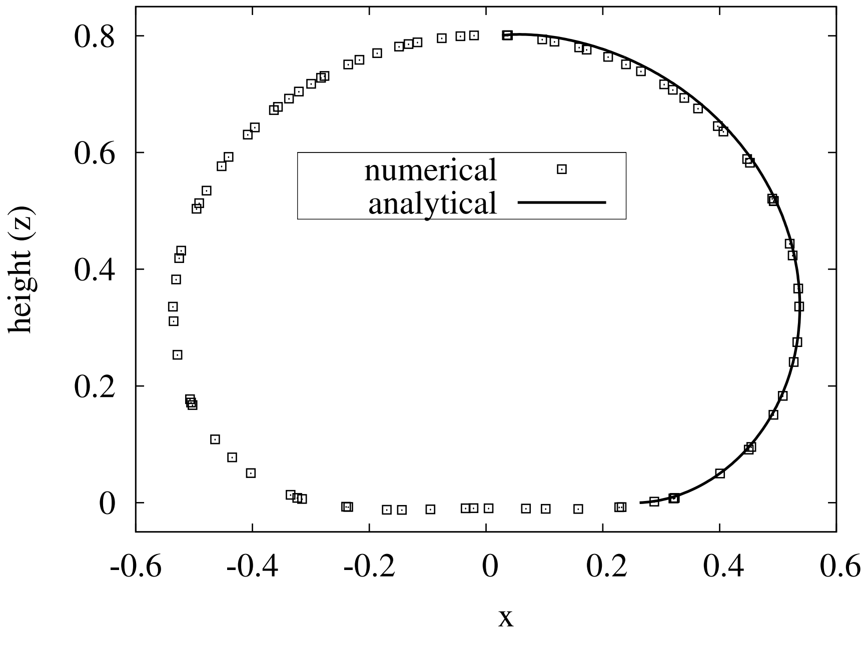


Figure 7.6: Comparison between the numerical solution of an axisymmetric sessile drop and the analytical solution of its shape derived by the Young-Laplace equation of capillarity.

with the frequency given by:

$$w^2 = \frac{24\sigma}{(3\rho_{in} + 2\rho_{out})R^3} \tag{7.9}$$

and an amplitude decay given by:

$$a(t) = a_0 e^{-t/\tau} \tag{7.10}$$

where ρ_{in} stands for the drop density, ρ_{out} represents the density of the surrounding fluid and R is the unperturbed drop radius. In the amplitude equation, a_0 is the initial drop amplitude that should be small enough to avoid the growth of undesired non-linear modes. The time is given by t and $\tau = R/5\nu$ where R is the unperturbed drop radius and ν is the kinematic viscosity. For this simulation a_0 was set to $0.1R$, assuring the requirements of the linearization process.

The non-dimensional parameters used for this simulation were $\sigma = 1$ and $R = 0.5$ as the drop's radius, $\rho_{in} = 1.0$, $\mu_{in} = 1.0$ for the drop and $\rho_{out} = 0.001$ and $\mu_{out} = 0.001$ for the external fluid. The domain limits were set to be $8D$x$8D$x$8D$. The coarse mesh had approximately 43000 tetrahedrons and 9100 nodes. The interface mesh had approximately 8900 triangles, from which 1800 triangles belong to the interface mesh. Instead, the refined mesh had approximately 60000 tetrahedrons and 11600 nodes, while the surface mesh had approximately 11700 triangles with 4100 triangles as part of the interface mesh.

Figure (7.7) shows the numerical solution of the interface's axial position for two different mesh refinement levels, namely coarse and refined. A comparison is then made with the analytical curve given by:

$$y(t) = y_0 + a_0 e^{-t/\tau} cos(wt). \tag{7.11}$$

In the above equation, y_0 corresponds to the initial oscillating surface's axial position. As expected, the predicted curve for the refined mesh has a better agreement to the analytical solution when compared to the coarse mesh. Moreover, the numerical oscillating frequency found for the refined mesh shows very good agreement to the analytical solution with an error $< 1\%$ while the coarse mesh presents an error of 4%.

Figure (7.8) shows the simulated drop at two different times with the inversion of the velocity direction in the z-axis. The simulations were performed with the re-meshing technique describe in Chapter 5 for the inner and outer mesh as well as for the surface mesh. Thus, it shows that the code is able to capture correctly the dynamics of capillary waves.

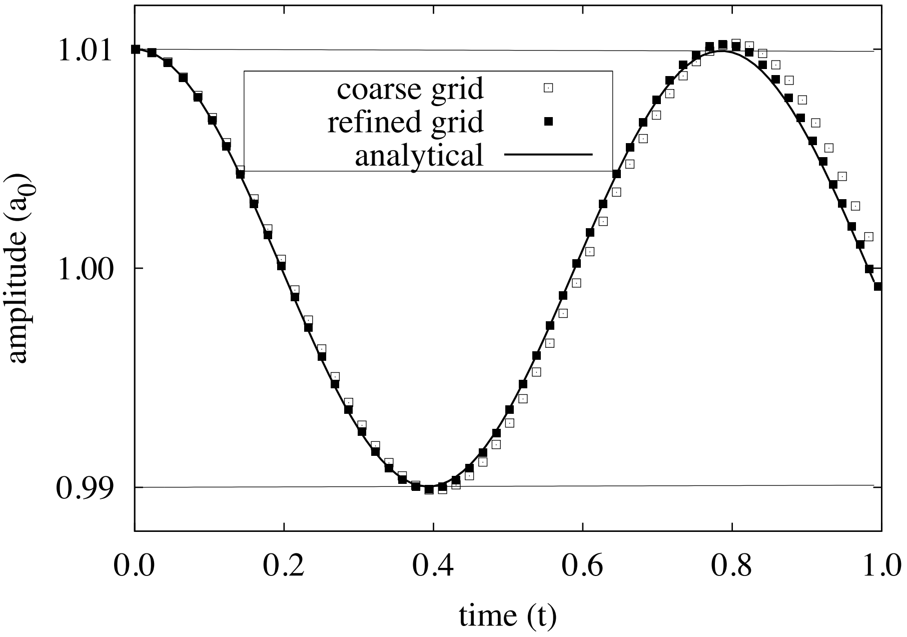
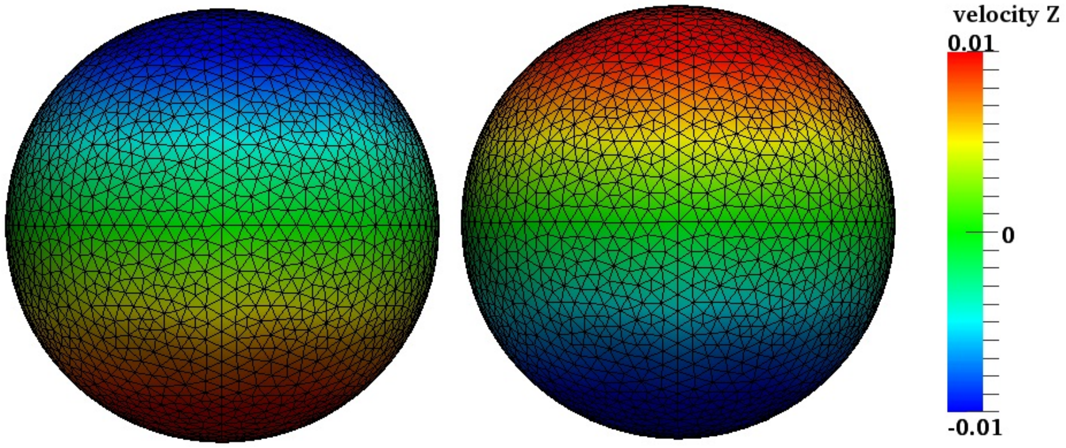


Figure 7.7: Drop oscillation amplitude. Comparison between numerical and analytical solution for two levels of mesh refinement. The analytical period is 0.785 and the decay rate is shown by the envelope represented by the lines below and above the oscillating curve. The oscillating period in the coarse mesh was found to be 0.820 for the coarse mesh while that for the refined mesh was 0.783.

7.5 Falling Drop in an Inert Media

The falling drop test case aims to compare the effects of the implemented convection, viscous and gravity terms versus the surface tension force. This dynamic test is set up by releasing a spherical drop with high surface tension, and letting it fall due to gravity. The position of the center of mass of the free falling non-deformable object yields an analytical solution which is given by a parabolic second degree polynomial equation and it can be used to compare to our simulation. Two simulations were performed keeping all the parameters constant but varying the viscosity ratio μ_{in}/μ_{out}, thus changing significantly the outside fluid resistance and consequently illustrating this effect with respect to the non-viscous analytical solution. A parallelepiped of $6D$x$6D$x$8D$ with the higher dimension along the direction of gravity was used. The parameters used in this simulation were $R = 0.5$ for the drop radius, $\rho_{in}/\rho_{out} = 1000$ as the density ratio, and $\mathrm{N} = 1000^{1/2}$ and $\mathrm{Eo} = 1$ for the *Archimedes* and *Eötvös* numbers, respectively. The sphere was initially placed on the upper region of the $4D$x$4D$x$8D$ domain, where D stands for the sphere diameter and discretized by approximately 30000 tetrahedrons and 5200 nodes. The interface mesh had approximately 1100 triangles and the total surface mesh had approximately 3000 triangles.

Figure (7.9) shows the simulated drop center of mass positions in comparison with the analyti-



(a)

(b)

Figure 7.8: Inversion of the velocity direction in the z-axis. (a) The top and bottom parts are squeezing the drop, and it corresponds to $t \approx 0.0$. (b) The velocity is inverted, thus the drop is being stretched in the z-axis, $t \approx 0.4$

cal solution given by $z = z_0 - (1/2)gt^2$, and Fig. (7.10) depicts the drop velocities compared to the analytical solution, given by $w = w_0 - gt$, where $z_0 = 5.5D$ and z represent the drop initial and final center of mass position, w_0 and w are the velocity along the z-coordinate, g stands for gravity and t is the non-dimensional time. As can be seen, the solution of the simulation for the low viscosity closely follows the analytical solution. According to Eq. (6.5) the mesh parameters used in all cases were $\beta_1 = 0$, $\beta_2 = 1.0$, $\beta_3 = 0.1$, $\gamma_1 = 0.1$ and $\gamma_2 = 0.2$. These results show that the convection and gravity terms are well balanced in the presence of the surface tension force; and that as the viscosity is decreased, the solution approaches the analytical solution.

7.6 Spherically growing bubble in a superheated liquid

To validate the proposed mass transfer model, a spherical vapor bubble is simulated in a superheated liquid with the fluid properties shown in Table (7.3). The numerical domain was set as a cubic shape and an outflow condition was imposed at all the walls, thus allowing the bubble to expand equally to all directions. Initially, a vapor bubble with its temperature equal to the saturation temperature T_{sat} is placed on the middle of the domain. As mentioned in Chapter (3), the surface temperature is set to the saturation temperature, and thus the only difference in temperature is due to the superheated liquid surrounding the vapor bubble. Considering that the temperature is non-dimensionalized such that $T^* = (T - T_s)/(T_w - T_s)$, the saturation temperature is set to $T_s = 0$ and the temperature of the superheated liquid is set to $T = 0^+$. The superheated liquid at the given saturation pressure starts to evaporate, thus the total liquid volume decreases and the vapor volume increases. At this point, an analytical

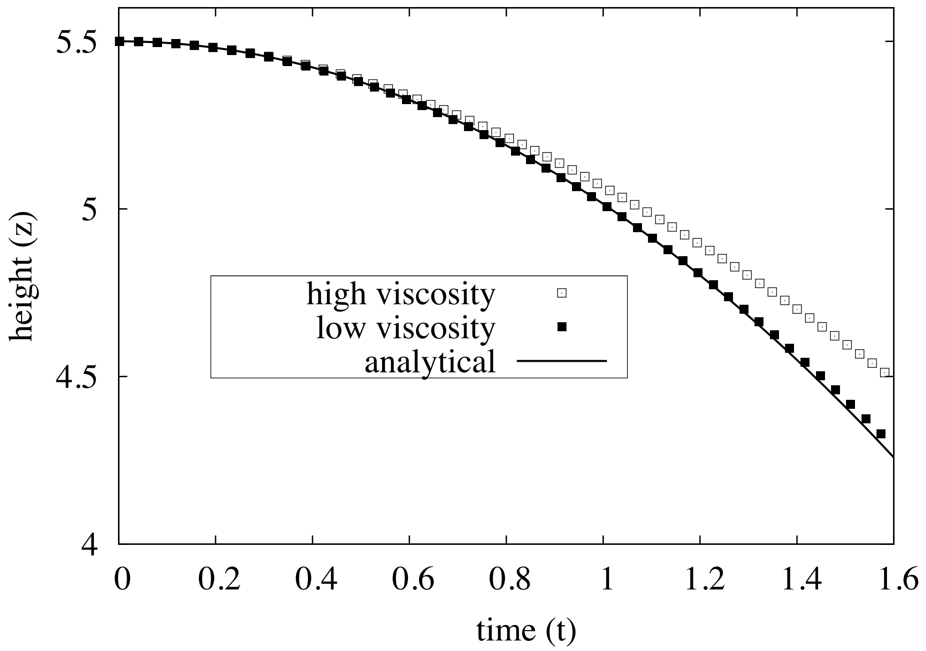


Figure 7.9: Trajectory of the falling drop immersed in different fluids. The drop trajectory of the high viscosity fluid (black squares) tends to deviate as expected from the analytical solution while the low viscosity fluid trajectory almost matches the free fall equation.

Table 7.3: Fluid properties

fluid	vapor phase				liquid phase				interface
	ρ $[kg/m^3]$	μ $[\mu Pa\cdot s]$	k $[W/mK]$	c_p $[J/kgK]$	ρ $[kg/m^3]$	μ $[\mu Pa\cdot s]$	k $[W/mK]$	c_p $[J/kgK]$	σ $[N/m]$
R134a	37.54	12.04	0.0173	1065	1187	185.4	0.079	1446	0.0074

solution can be used to compare the change of vapor volume with time. Such an expression is given by:

$$R(t) = 2\beta\sqrt{(kt)} \tag{7.12}$$

where R is the bubble's radius, k is the thermal conductivity and β is a constant calculated in [80]. Note that the viscous and surface tension effects are neglected during the simulation, thus Eq. (7.12) is valid for the growth of the vapor bubble.

Figure (7.11) shows the growth of the vapor bubble with time due to evaporation process. From

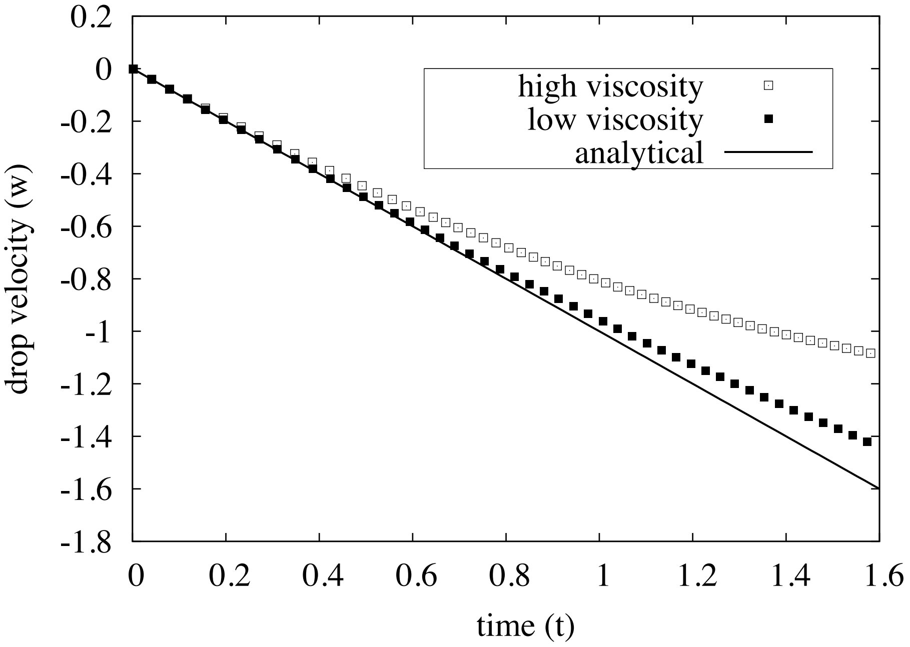


Figure 7.10: Drop's fall velocity driven by gravitational effects. The effects of deacceleration are stronger when the inert media has a high viscosity compared to the analytical solution of the free fall velocity profile, while in the low viscosity fluid, the deacceleration is more pronounced, obstructing the descent of the drop.

$t \approx 0.02$ to $t \approx 0.12$, the vapor bubble expands linearly due to heat conduction. From $t \approx 0.13$ to the end of the simulation, the bubble asymptotically reduces its expansion and begins to deviate from the analytical solution, due to different assumptions used to derive Eq. (7.12) with respect to those in the numerical simulation. Figure (7.12) depicts the three velocity components showing the vapor bubble expansion. As can be seen, the velocity increases with time as well as the bubble radius.

Further investigation of the boiling process in the present two-phase code is still required, looking at additional cases and flows.

7.7 Rising Bubble

A comparison with experimental data has also been carried out to validate the code. In this test case, the difference of fluid properties (viscosity and density) as well as all the terms of the momentum and continuity equations are tested simultaneously. Additionally, the resilience of the re-meshing procedure implemented in this work is massively tested due to the significant change in bubble shape, where the final bubble geometry presents a dimpled ellipsoidal-cap form (see classification in [37]). As can be seen in Figs. (7.13-7.17), the distortion of the surface mesh at the bottom of the bubble becomes larger as the viscosity decreases, thus making the re-meshing process even more necessary.

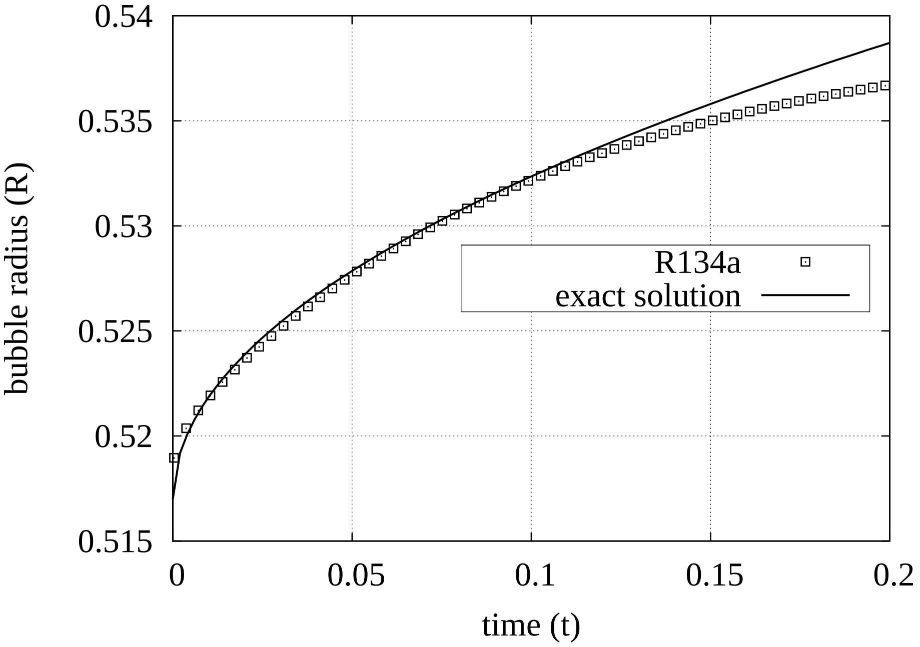
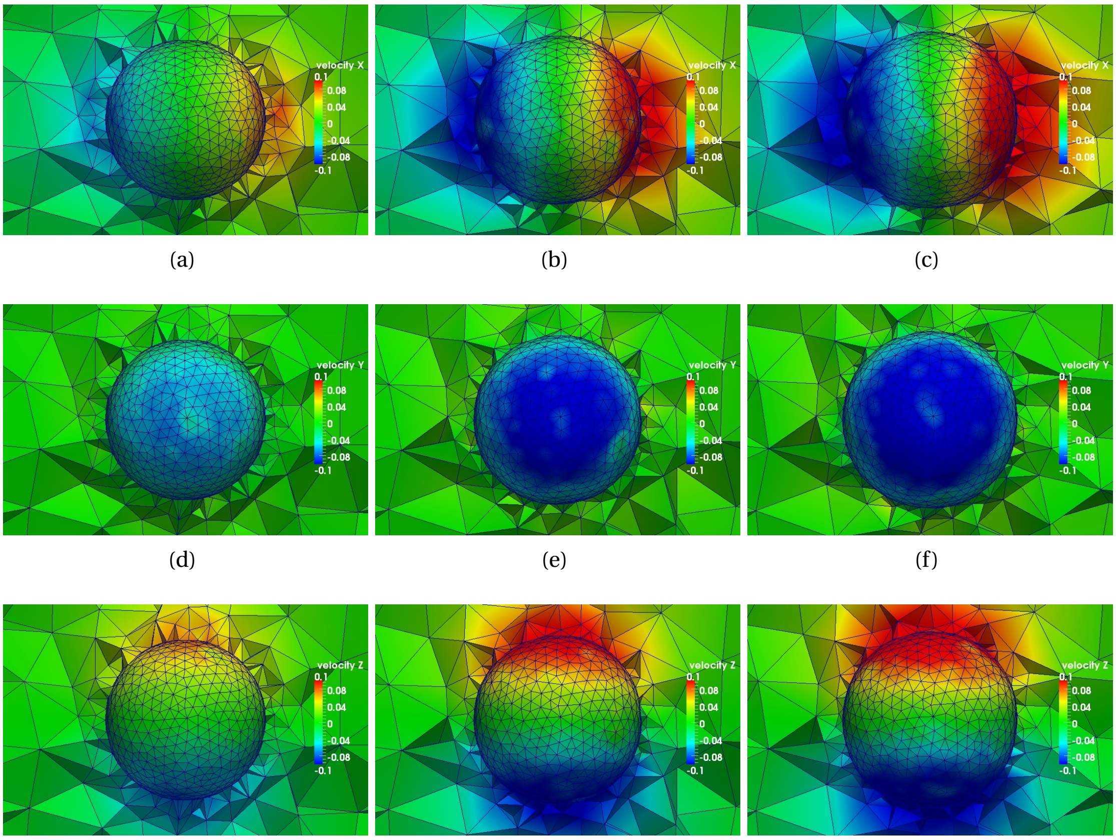


Figure 7.11: Growth of a vapor bubble due to evaporation.

The numerical results were compared to the widely cited experiments performed by [11]. Tests were carried out to predict the terminal velocity of a rising air bubble in aqueous sugar solutions for five different viscosities. According to the experiments, the volume of the generated bubbles was 9.3 cm^3, thus the diameter is $d = 2.61$cm. The surface tension of 0.078N/m was that of the air-water-sugar interface, the bubble air's viscosity and density are 0.0000178kg/ms and 1.225kg/m^3 respectively. We considered an average value for the aqueous solution density to be 1350kg/m^3, since the experiments presented measurement variations, and five different liquid viscosities $\{2.73, 1.28, 0.54, 0.28, 0.13\}$kg/ms, thus changing the final shape of the rising bubbles. For the first three cases presented, two different mesh refinement levels were used, namely coarse and refined. The former had approximately 5000 volumetric nodes, 33000 tetrahedrons, 1400 surface nodes and 2800 surface triangles, and the later had approximately 18000 volumetric nodes, 111000 tetrahedrons, 4000 surface nodes and 8000 surface triangles. For the last two cases, only the refined mesh was used. The geometry of the domain consisted in a parallelepiped with dimensions of $8D$x$8D$x$15D$ with the higher dimension along the gravity direction was used and a bubble with radius $R = 0.5$ was placed in the bottom of the domain where the center of mass is located at $z = 3$.

Figure (7.13) shows bubble shape transition and the center of mass velocity of the rising air bubble immersed in the most viscous solution. It can be seen that at time $t = 1.9$ the bubble



(a) (b) (c)

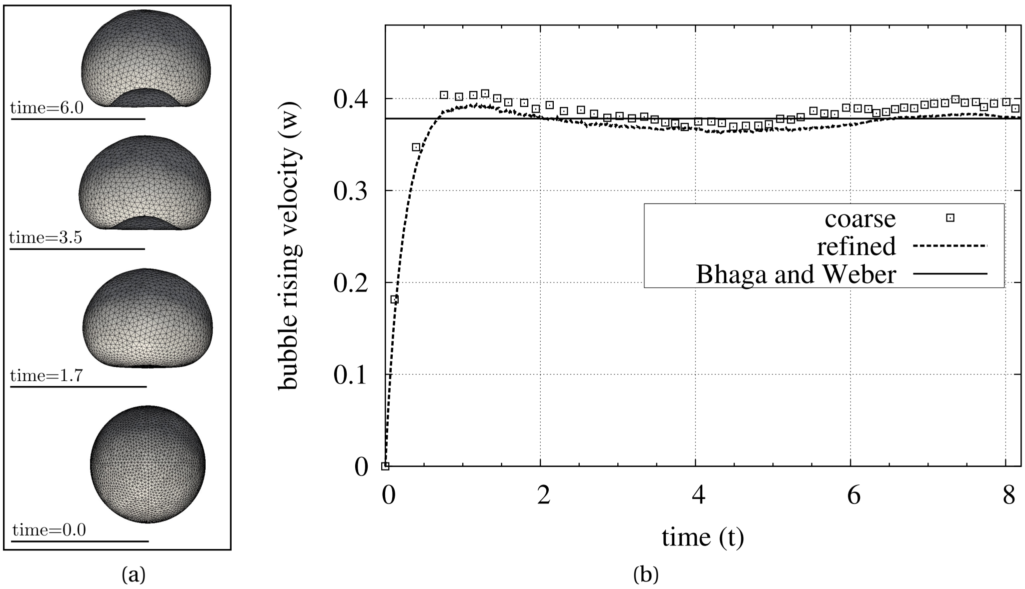
(d) (e) (f)

(g) (h) (i)

Figure 7.12: Velocity components of a vapor bubble growing due to evaporation of the superheated liquid. (a)-(c) x-component , (d)-(f) y-component and (g)-(i) z-component.

velocity reaches its respective terminal velocity. The final bubble shape is reached at $t \approx 5.0$. The bottom edge of the bubble, where the curvature changes its sign, is not so pronounced due to the slow ascension velocity and thus the mesh can be regularize with relative ease. The mesh parameters used in these simulations were $\beta_1 = 0.0$, $\beta_2 = 0.8$, $\beta_3 = 0.1$ for the volumetric mesh and $\gamma_1 = 1.0$ and $\gamma_2 = 0.1$ for the surface mesh. The non-dimensional parameters were set to $Eo = 116$, $N = 194.88$, $\rho_{in}/\rho_{out} = 0.0009$, $\mu_{in}/\mu_{out} = 6.53E-06$.

Changing the concentration of the aqueous sugar solution and thus its viscosity from 2.78 to 1.28kg/ms, the Morton number changes from $Mo = 848$ to $Mo = 41.1$ and a higher bubble ascension velocity is obtained as can be seen in the Fig. (7.14). The bubble's final velocity is reached at time $t = 1.7$, and then it remains nearly constant until the end of the simulation. The final bubble shape is reached at $t \approx 5.2$. Note that due to the higher ascension velocity profile and the consequently larger variation of the bubble shape, it was observed that both the volumetric and surface meshes should adapt faster to the onset of non-desired low quality elements. However, this requirement may be reduced by increasing the smoothing parameters



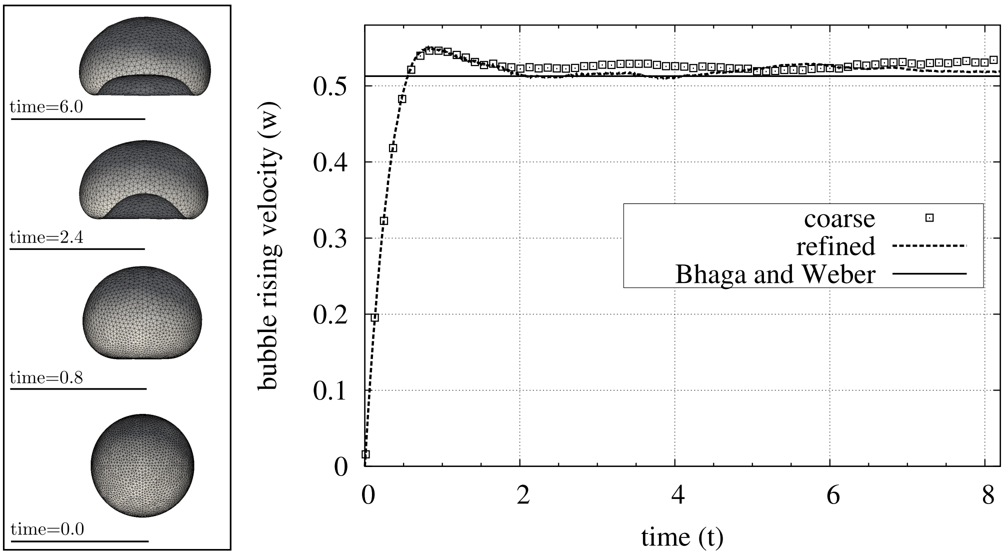
(a) (b)

Figure 7.13: Rising of an air bubble immersed in the aqueous sugar solution with the highest viscosity $\mu = 2.73$. (a) Bubble's shape evolution. (b) Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional.

β and γ so that the meshes are regularized, thus avoiding the collapse of nodes and edges near the bubble. The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\beta_3 = 0.3$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.2$. In this simulation, the non-dimensional parameters were set to $Eo = 116$, $N = 194.88$, $\rho_{in}/\rho_{out} = 0.0009$, $\mu_{in}/\mu_{out} = 1.39E-05$.

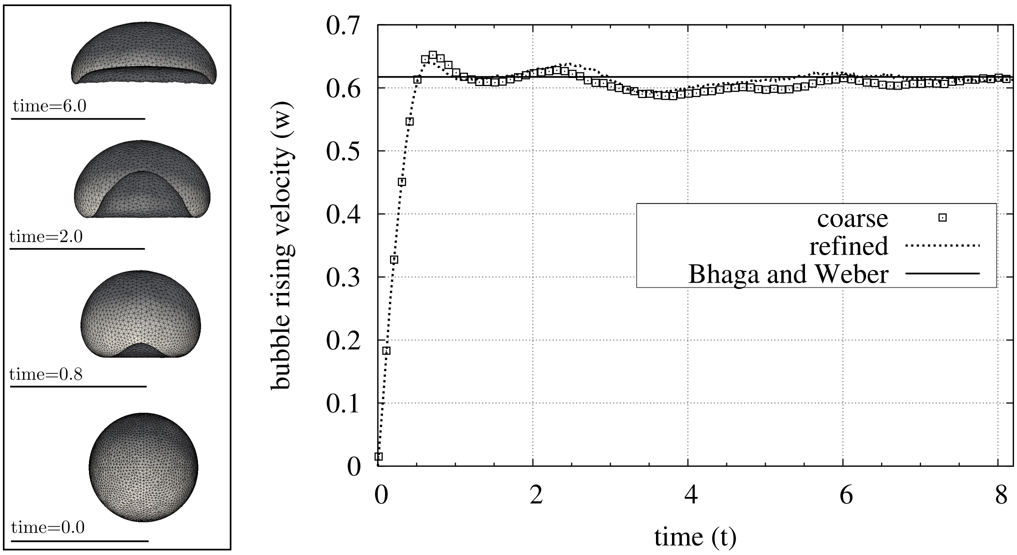
Figure (7.15) shows the rising of an air bubble immersed in the third viscous solution (0.54kg/ms and Mo = 1.31). At time $t = 1.0$ the bubble velocity reaches its respective terminal velocity, but its shape varies continuously up to time $t \approx 3.8$ when the bubble shape is nearly stable. The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.2$. The volumetric mesh parameter was set higher than the previous case to $\beta_3 = 0.5$, thus producing a suitable distribution of mesh nodes even in the presence of higher velocity gradients. In this simulation, the non-dimensional parameters were set to $Eo = 116$, $N = 1091.57$, $\rho_{in}/\rho_{out} = 0.0009$, $\mu_{in}/\mu_{out} = 3.29E-05$.

Figure (7.16) shows the rising of an air bubble immersed in the fourth viscous solution (0.28kg/ms and Mo = 0.103). Note that for this case and the next one, only the refined mesh was used. At time $t = 3.7$ the bubble velocity reaches its respective terminal velocity, but its shape varies continuously up to time $t \approx 5.5$ when the bubble shape is nearly stable. The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.2$. The volumetric mesh parameter was set higher than the previous case to $\beta_3 = 0.8$, thus producing a suitable distribution of mesh nodes even in the presence of higher velocity gradients. The



(a) (b)

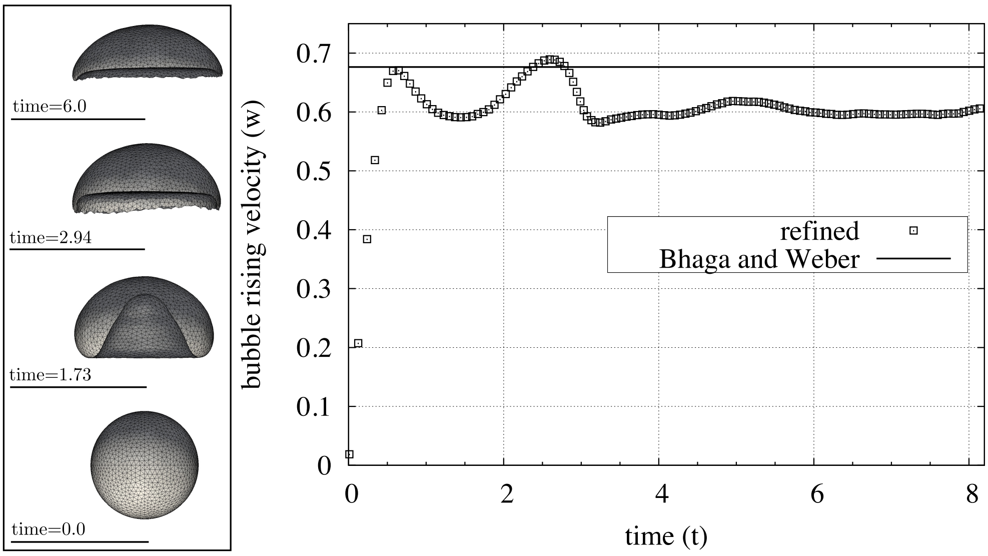
Figure 7.14: Rising of an air bubble immersed in the aqueous sugar solution with moderate viscosity $\mu = 1.28$. (a) Bubble's shape evolution. (b) Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional.



(a) (b)

Figure 7.15: Rising of an air bubble immersed in the aqueous sugar solution with viscosity $\mu = 0.54$. (a) Bubble's shape evolution. (b)Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional.

sharp edges found in this simulations demanded severe surface remeshing, thus the excessive deletion of nodes and elements introduced numerical diffusion that affected the final bubble's velocity. However, the final bubble's shape is in agreement with the experiments. In this simulation, the non-dimensional parameters were set to $Eo = 116$, $N = 3892.86$, $\rho_{in}/\rho_{out} = 0.0009$, $\mu_{in}/\mu_{out} = 6.23E-05$.



(a) (b)

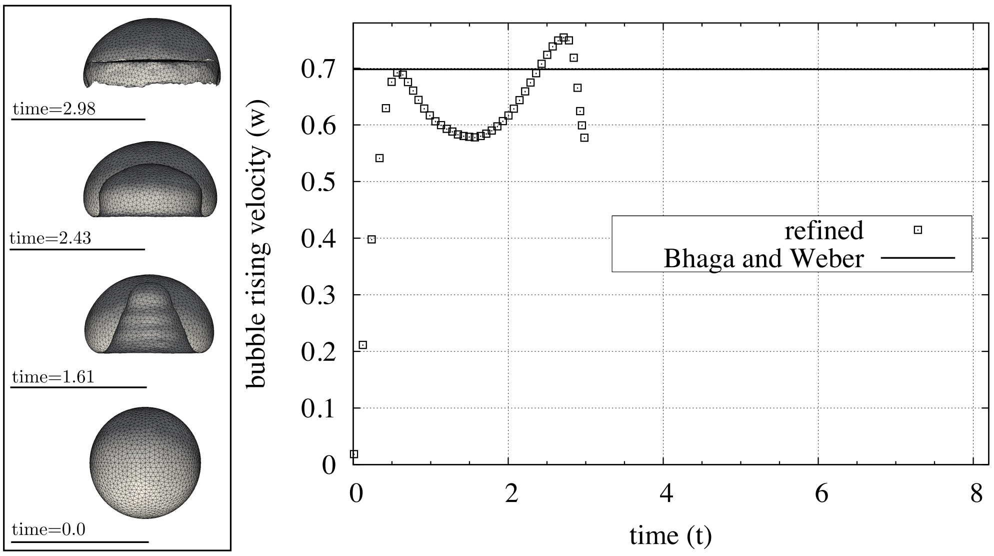
Figure 7.16: Rising of an air bubble immersed in the aqueous sugar solution with viscosity $\mu = 0.28$. (a) Bubble's shape evolution. (b)Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional.

Figure (7.17) depicts the rising of an air bubble immersed in the least viscous solution (0.13kg/ms and Mo = $4.63E-3$). Due to the pronounced shape deformation, the simulation did not reach the final stage. At time $t = 2.98$, the bottom

Despite the geometrical handling of the surface mesh, the bubble oscillated near the terminal velocity as reported in the experiments, and therefore it can be conclude that the bubble dynamics and the difference of fluid properties are well represented by the proposed model, as can been seen in Fig. (7.17b).

Different mesh parameters have been tested to handle the large surface distortions produced by this simulation condition. The most suitable was found to be $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.2$. The volumetric mesh parameter was set higher than the previous case to $\beta_3 = 0.9$. In this simulation, the non-dimensional parameters were set to $Eo = 115$, $N = 18124.01$, $\rho_{in}/\rho_{out} = 0.0009$, $\mu_{in}/\mu_{out} = 13.44E-05$.

For the simulations performed above, a small fluctuation was observed when the bubble reached its final shape, as been reported by other numerical benchmarks found in the litera-



(a) (b)

Figure 7.17: Incomplete rising of an air bubble immersed in the aqueous sugar solution with the least viscosity $\mu = 0.13$. (a) Bubble's shape evolution. (b)Bubble's center of mass velocity. The straight solid line represents the terminal velocity found by [11]. Velocity and time are non-dimensional.

ture. Such a behavior does not seem to affect the final solution. Moreover, the flexibility given by the mesh control parameters β and γ allows an easy maintenance of the computational mesh, achieving the representative elements with satisfactory shape, thus increasing the accuracy of the Finite Element method. The required mesh treatment for the last case reported ($\mu = 0.13$)) still needs to be further investigated.

7.8 Rising of Taylor Bubbles

In this section, a more challenging test has been carried out to validate the code against the terminal velocity and liquid film thickness of elongated bubbles in circular channels. The wall effects are included in the bubble dynamics and the results are compared to the well known flow pattern map of [97] and the Brown's theoretical solution for the film thickness [14]. The modeling of such flows increases significantly the obstacles of the remeshing process, since the formation of the thin liquid film between the bubble and wall requires a fine mesh to capture the flow mechanisms. Again, the flexibility of the Finite Element method is explored in the development of the computational meshes, in which different mesh element sizes are used in the 3-dimensional domain.

Figure (7.18) shows the flow pattern map of White and Beardmore for air bubbles in circular channels. To characterize the rising velocity of air bubbles in different solutions, 7 regions

Table 7.4: Fluid properties

fluid	properties			dimensionless number		film thickness
	ρ $[kg/m^3]$	μ $[\mu Pa \cdot s]$	σ $[mN/m]$	Eo [-]	Mo [-]	δ [-]
sucrose 1	1.172	5.650	77.7	40	1E-07	0.0617
sucrose 2	1.172	5.650	77.7	100	1E-07	N.A.
glycerol	1.260	712.0	63.1	40	1E-01	0.1483
diluted glycerol	0.234	154.0	64.8	100	1E-02	0.1196
sugar syrup 1	1.420	20900	77.2	400	1E+04	0.2267
sugar syrup 2	1.420	20900	77.2	3	1E+06	N.A.
air	1.789	1.225	—	—	—	—

were identified and related to viscous, interfacial and inertial forces. The solutions used in the experiments varied from distilled water to sugar syrup, glycerol, oil, sucrose and ethanol. Table 7.4 summarizes the fluid properties used in the rising Taylor bubble simulations and the dimensionless film thickness δ calculated from the following expression:

$$\delta = \left[\frac{3}{2} Fr \frac{Mo^{1/4}}{Eo^{3/4}} (R - \delta) \right]^{1/3} \tag{7.13}$$

where Fr, Eo and Mo are the dimensionless numbers defined in Chapter (3) and R is the channel radius. This expression is derived from [14] and calculates the stable film thickness of a rising bubble. However, the stability of the film thickness requires that the bubble length L_b must be large enough so that the uniform film can be measured. The initial bubble's length L_b in all simulations was set to be $L_b > 2D$. Three fluids with different properties were used, namely sucrose, glycerol and sugar syrup, as according to the experimental database. Doing so, the code can be tested and benchmarked to different zones found in the flow pattern of Fig. (7.18).

The numerical solution for the rising bubble requires a long domain to be compatible to the experiments. According to [15], the development of the bubble's shape and, consequently, the terminal velocity requires that the numerical domain should be $8D$ long in the gravity direction. As mentioned before, an extra mesh refinement is needed to solve the liquid film formed between the wall and the confined bubble, thus the total number of mesh elements increases dramatically. Considering that the simulations are performed in 3-dimensions and the experimental test section is constant in the gravity direction, a moving referential frame technique is employed to shorten the numerical domain, thus allowing a faster computation. Figure (7.19) depicts the schematical representation and the boundary conditions used to

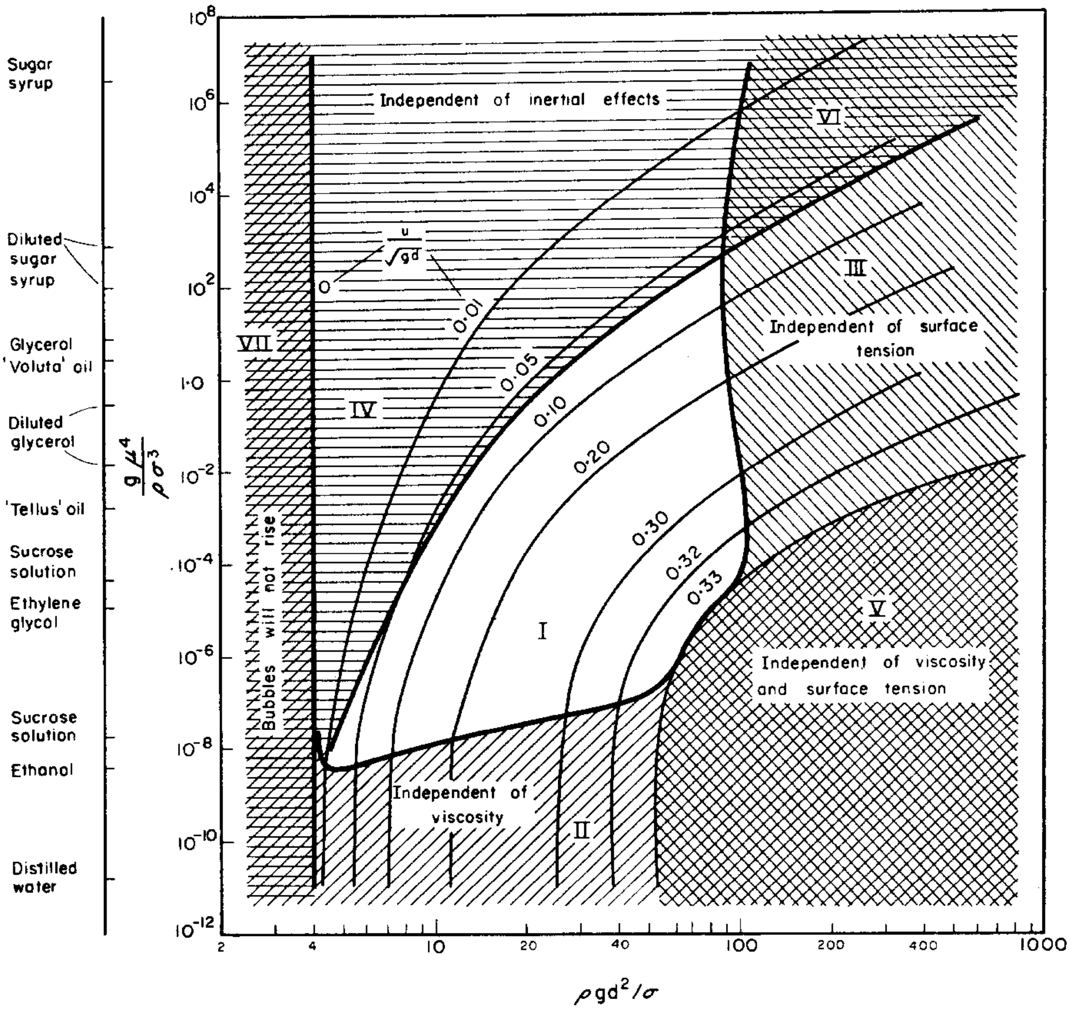
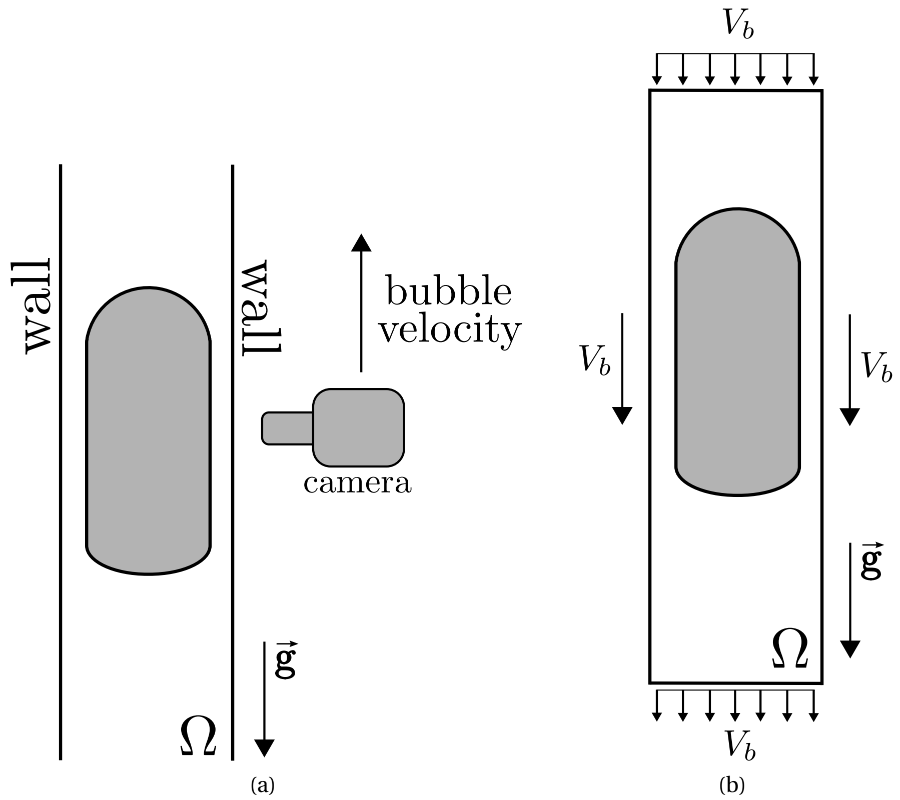


Figure 7.18: Flow pattern map of rising bubble in cylindrical vertical tubes [97]. 7 regions were identified to characterize the dependency of the velocity to viscous, interfacial and inertial forces. The horizonta axis stands for the *Eötvös* number and the vertical axis to the *Morton* number. The curved lines are dimensionless velocity, which is equivalet to the *Froude* number.

represent the referential domain. In Fig. (7.19a) the bubble moves upward with velocity V_b and the camera tracks the bubble's motion with the same velocity. By mean of relative velocity, if V_b is removed from the bubbles velocity, then the bubble and the camera will be steady and all the remaing objects will move downward with a velocity equal to $-V_b$. Consequently, the boundary conditions is imposed in such a way that the fixed bubble should "feel" the shear from the wall. This is represented by Fig. (7.19a), where the inflow and outflow conditions are imposed in the top and bottom wall with respect to the bubble's velocity V_b, being the same applied to the side walls.

The numerical domain used to simulate all the fluids given by Table (7.4) was set to $D \times 5D + L_b$,



(a) (b)

Figure 7.19: Schematical representation of the moving frame technique, which uses the principles of relative velocity to shorten the numerical domain. (a) The rising bubble moves with velocity V_b and the camera tracks its motion with the same velocity. (b) The boundary conditions applied to simulate the same condition illustrated in (b) requires that the walls move downward with velocity V_b.

where D stands for the circular channel diameter and L_b is the bubble length. The bubble was placed far from the bottom part of the domain to guarantee that the bubble's wake is well captured, thus allowing the bottom of the bubble to deform. Figure (7.20) shows the front, the back and the side views of the computational domain used at all simulations performed in the rising Taylor bubble test. As can be seen, the mesh is more dense at the wall and close to the bubble, where the maximum edge length size allowed is $h_{max} = 0.02$. Below the bubble, a larger value of h is allowed, however the maximum edge length admissible should be able to capture the wake behind the bubble, thus it was set to be $h_{max} = 0.05$. Finally, in the top part of the domain, the maximum edge length was set to $h_{max} = 0.09$. Such a mesh refinement reduces considerably the computational time if compared to an uniform grid. Moreover, during the simulation, nodes are added and deleted according to the description of mesh adaptive refinement in Chapter 6, assuring accurate results.

Figure (7.21) shows the time progression of a Taylor air bubble immersed in a sugar syrup

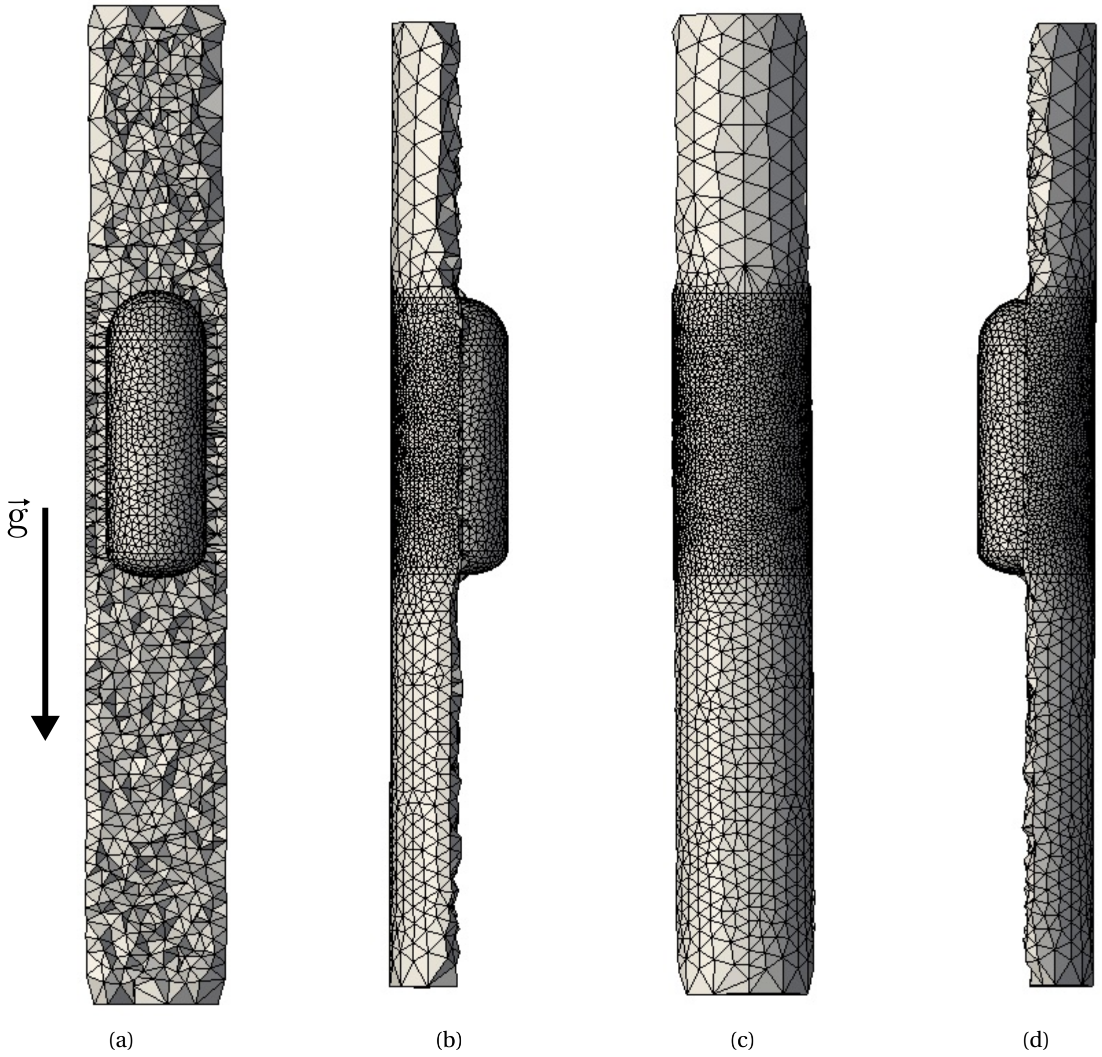


Figure 7.20: Tetrahedron mesh used to simulate the rising Taylor bubble. The boundary mesh is more refined close to the bubble to capture the mechanisms of the thin liquid film. Above the bubble, the mesh is less refined and behind the the bubble a fine mesh is used to capture the bubble's wake.

solution. The bubble is initialized as a "bullet" shape to reduce the initial perturbation generated by the initial bubble form. The initial film thickness was set to $\delta_o = 0.37$ and the mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.1$. The air bubble deformation remains with rounded end, and not presenting sharp edges, thus the adaptive mesh refinement is able to handle with relatively ease. The bubble terminal shape was also compared to different numerical works and it is qualitatively in good agreement. In Fig. (7.22) is shown the evolution in time of the bubble's center of mass velocity, in which the computed terminal velocity approaches the value found in the flow pattern map. The error was found to be < 0.1%.

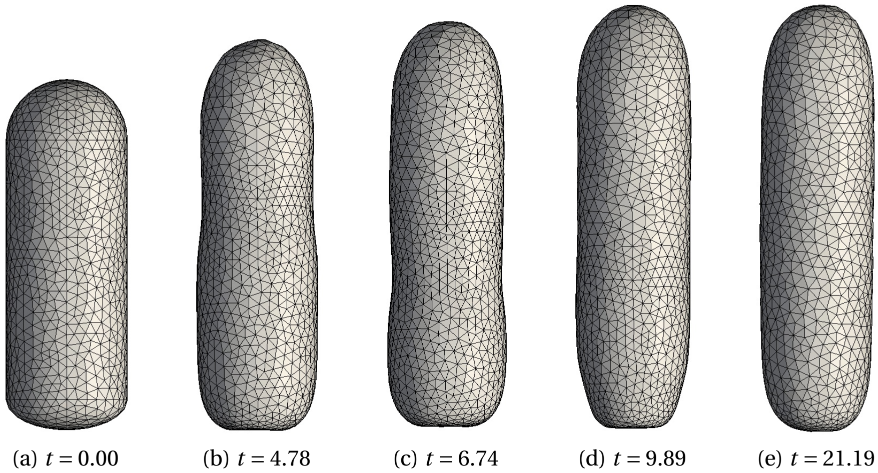

(a) $t = 0.00$ (b) $t = 4.78$ (c) $t = 6.74$ (d) $t = 9.89$ (e) $t = 21.19$

Figure 7.21: Bubble shape evolution with time for an air bubble in a sugar syrup solution with dimensionless numbers $Mo = 10^4$, $Eo = 400$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 21.19$.

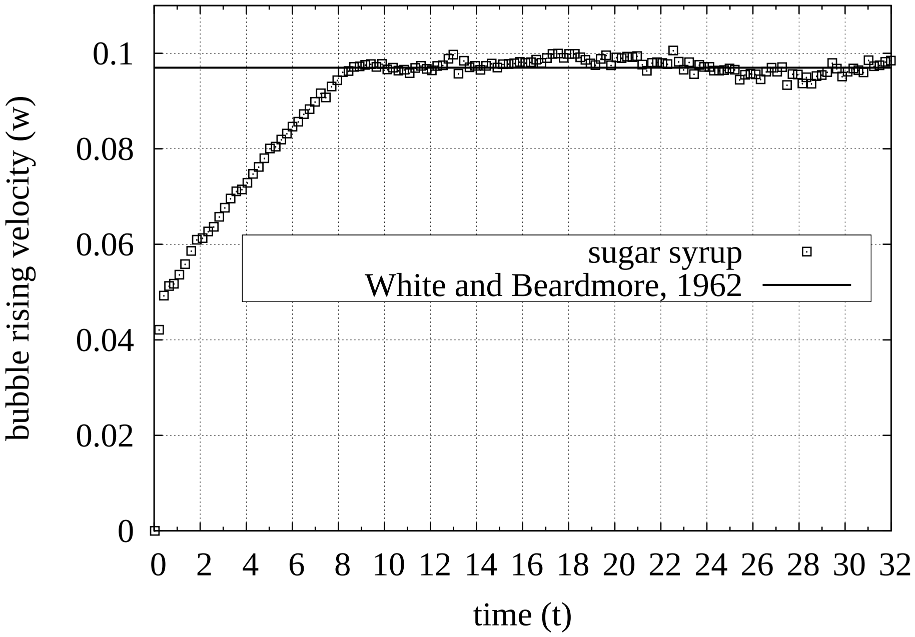
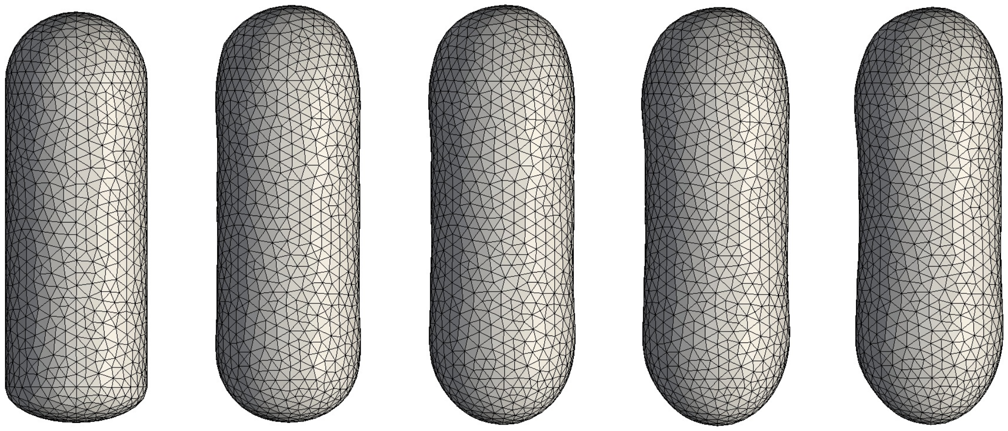


Figure 7.22: Rising of an air Taylor bubble immersed in the sugar syrup solution with dimensionless numbers set to $Mo = 10^4$ and $Eo = 400$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional.

In the test case presented by Fig. (7.23), it was observed that the bubble's shape converges to a symmetric shape. This is due to strong viscous force compared to the gravitational one, which does not allow the bubble to rise. In the numerical test, a small velocity flotation of the order of 10^{-3} was found, as can be seen in Fig. (7.24). In fact, it was observed that the simulated bubble did not rise, showing that the velocity flotation is due to the interface motion, which converges to its equilibrium shape. In this simulation, the initial film thickness was set to $\delta_o = 0.37$ and the mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.1$.



(a) $t = 0.00$ (b) $t = 2.50$ (c) $t = 4.83$ (d) $t = 7.07$ (e) $t = 9.27$

Figure 7.23: Bubble shape evolution with time for an air bubble in a sugar syrup solution with dimensionless numbers $Mo = 10^6$, $Eo = 3$. In such a condition, the bubble does not rise due to the stronger viscous force compared to the gravitational force. (a) Initial bubble shape with $t = 0$. (b-e) Bubble shape transient solution.

Figure (7.25) shows the time progression of a Taylor air bubble immersed in a sucrose solution. The same bubble shape and film thickness, as the previous cases, is used as an intial shape. In the transient evolution, the bubble's velocity reached its maximum velocity at time $t \approx 1$, and its terminal velocity at time $t \approx 3.7$ Also, it was shown that the bottom part of the bubble was pulled in and oscillated until convergence at $t \approx 7.4$ The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.1$ and the dimensionless numbers were set to $Mo = 10^{-7}$, $Eo = 40$. Figure (7.26 presents the transient solution of the bubble's center of mass velocity. It was observed, an overshooting of the ascension velocity from time $t = 0$ to $t = 1.1$, due to the initial deformation of the bottom part of the bubble, and consequently acceleration of the center of mass. The result agreed to the prediction of the flow pattern map, obtaining an error of 1.1%.

A difficult condition was tested in Fig. (7.27), in which the bubble surface changed dramatically. As can be seen at time $t = 2.70$, the bottom part of the surface mesh presents sharp edges due to the high curvature of the surface, thus shutting down the simulation. However the rising velocity in Fig. (7.28) shows an oscillating trend close to the solution given by the flow pattern

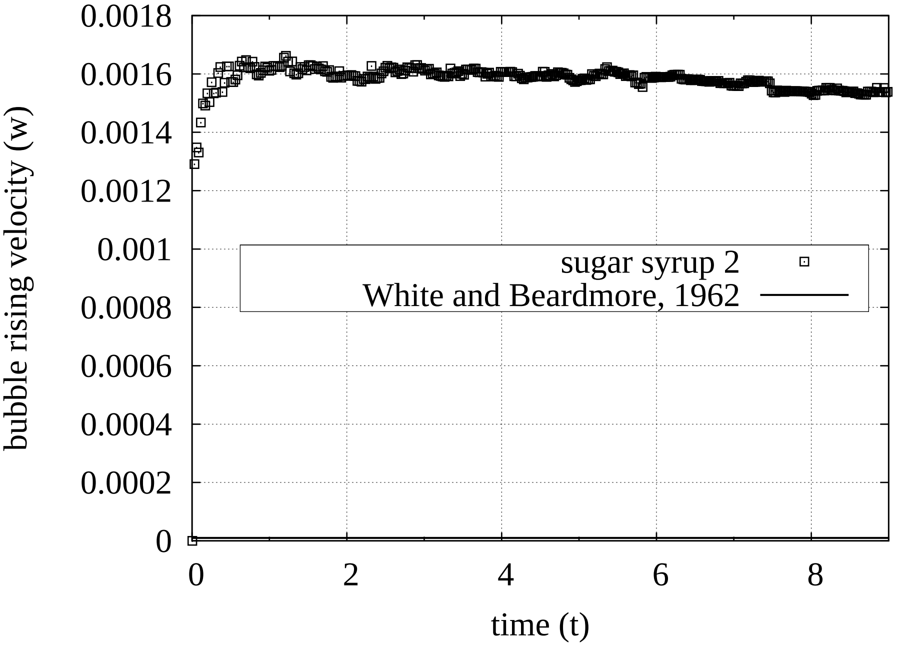
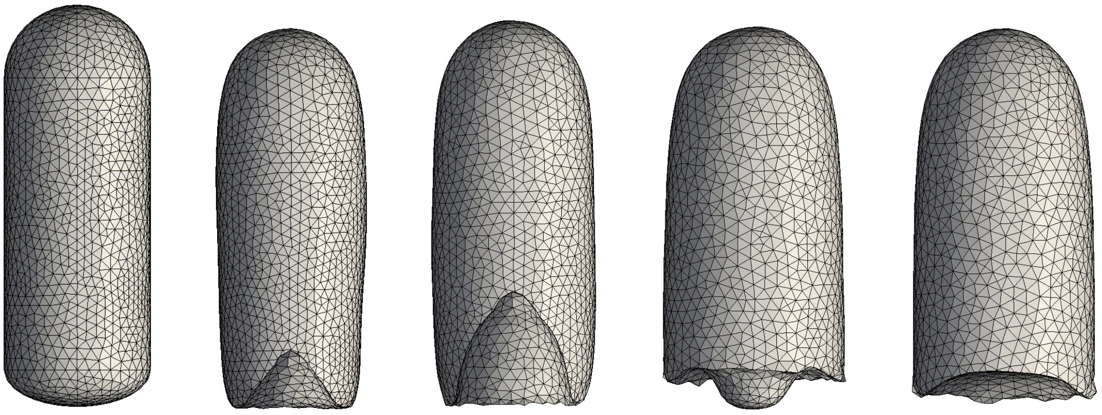


Figure 7.24: Rising of an air Taylor bubble immersed in the second sugar syrup solution with dimensionless numbers set to $Mo = 10^6$ and $Eo = 3$. According to [97], the terminal bubble's velocity is zero, thus the bubble does not rise. The numerical simulation shows that the bubble does not rise, however a small residual is found in the rising velocity w. Velocity and time are non-dimensional.



(a) $t = 0.00$ (b) $t = 1.09$ (c) $t = 1.78$ (d) $t = 3.05$ (e) $t = 7.41$

Figure 7.25: Bubble shape evolution with time for an air bubble in a sucrose solution with dimensionless numbers $Mo = 10^{-7}$, $Eo = 40$. The adaptive mesh refinement proposed in this work captures accurately the strong shape distortion produced by the high ascension velocity. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape change during transient solution. (e) Terminal bubble shape with $t = 7.41$.

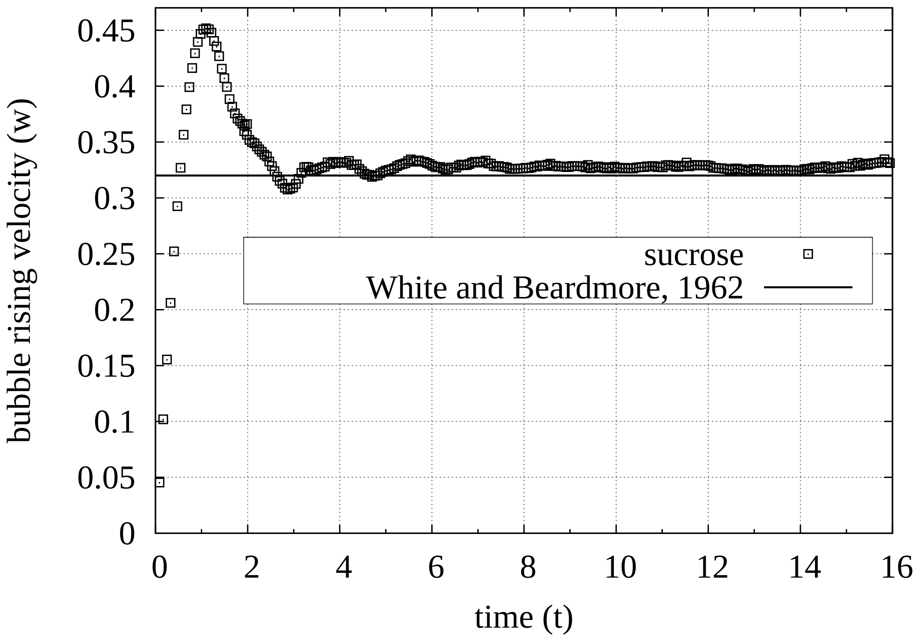


Figure 7.26: Rising of an air Taylor bubble immersed in a sucrose solution with dimensionless numbers set to $Mo = 10^{-7}$ and $Eo = 40$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional.

map. This indicates that, despite the difficulty in handling the surface mesh, the gravitational effects are well balanced with the inertial and interfatial effects.

Figure (7.29) shows the time progression of a Taylor air bubble immersed in a glycerol solution. The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 0.5$, $\gamma_1 = 0.5$ and $\gamma_2 = 0.1$ and the dimensionless numbers was set to $Mo = 10^{-2}$ and $Eo = 100$. In this case, the mesh parameters were slightly modified to test different mesh conditions in the simulation. The obtained result shows that the bubble shape and terminal velocity agrees well to experimental data. In Fig. (7.30) is shown the evolution in time of the bubble's center of mass velocity, in which the computed terminal velocity approaches the value found in the flow pattern map. The error was found to be 6.5%.

Figures (7.32 and 7.31) show the bubble shape and the rising velocity evolution with time, respectively. It was observed that the transient solution of the bubble shape presented round ends in the bottom part during the whole simulation. Moreover, a linear rising velocity was found in the interval $t = [1,4]$, followed by its convergence to the terminal velocity, this resembles the same rising velocity evolution of the sugar syrup solution. The error was found to be < 1.0%.

Table (7.5) shows the comparison of the obtained numerical liquid film thickness δ_{num} to the liquid film thickness calculated by Brown's correlation δ_{cor} given by Eq. (7.13). The film

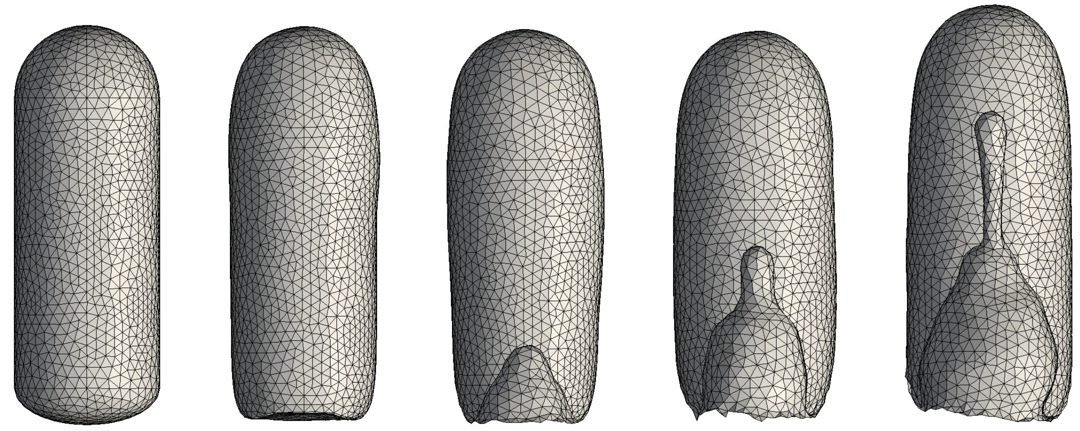

(a) $t = 0.00$ (b) $t = 0.71$ (c) $t = 1.17$ (d) $t = 1.76$ (e) $t = 2.70$

Figure 7.27: Bubble shape evolution with time for an air bubble in a sucrose solution with dimensionless numbers $Mo = 10^{-7}$, $Eo = 100$. The high ascension velocity and the strong deformation of the surface mesh shut the simulation down. Mesh analysis suggest that a specific surface mesh refinement is required to handle such a deformation. (a) Initial bubble shape with $t = 0$. (b-e) Bubble shape during transient solution.

thickness error δ_{error} is calculated according to the follow equation:

$$\delta_{error} = \sqrt{\frac{\sum(\delta_{cor} - \delta_{num})^2}{\sum(|\delta_{cor}|)^2}} \tag{7.14}$$

The good agreement of the liquid film thickness with the Brown's correlation indicates that the proposed method is suitable to describe the mechanisms of film formation between a vapor

Table 7.5: Dimensionless liquid film thickness

fluid	δ_{num}	δ_{cor}	δ_{error} [%]
sucrose 1	0.0681	0.0617	10.3
sucrose 2	0.0979	N.A.	N.A.
glycerol	0.1413	0.1483	4.72
diluted glycerol	0.1203	0.1196	0.58
sugar syrup 1	0.2190	0.2267	3.39
sugar syrup 2	0.1320	N.A.	N.A.

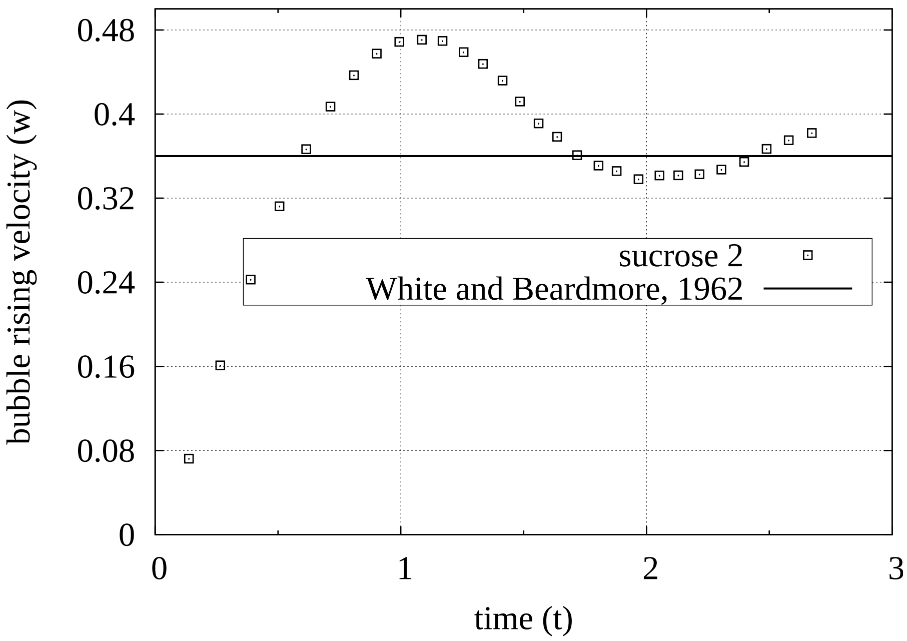
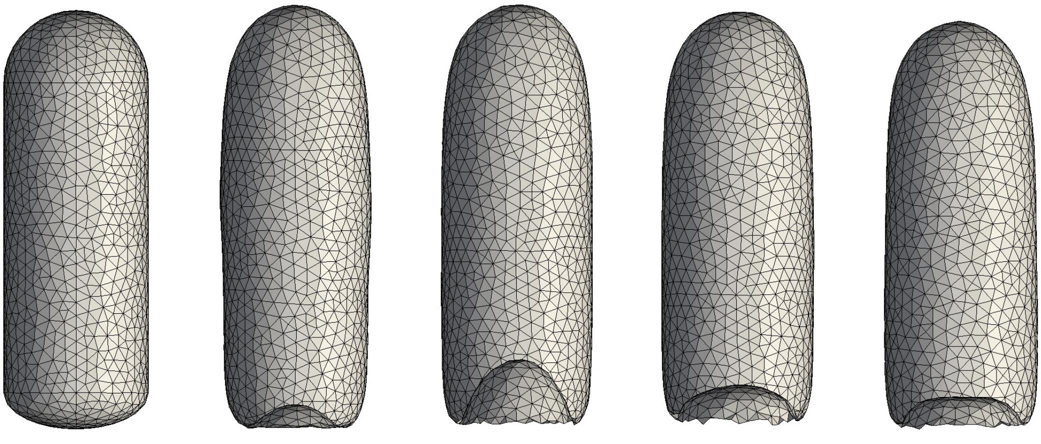


Figure 7.28: Rising of an air Taylor bubble immersed in a glycerol solution with dimensionless numbers set to $Mo = 10^{-7}$ and $Eo = 100$. The sharp edged produced by the high ascension velocity is not handled by the remeshing process. Mesh analysis suggested that a different edge length is required to model high curvature shapes, which is the case of the bottom part of the bubble. Velocity and time are non-dimensional.



(a) $t = 0.00$ (b) $t = 1.15$ (c) $t = 2.39$ (d) $t = 2.86$ (e) $t = 6.00$

Figure 7.29: Bubble shape evolution with time for an air bubble in a glycerol solution with dimensionless numbers $Mo = 10^{-2}$, $Eo = 100$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 6.00$.

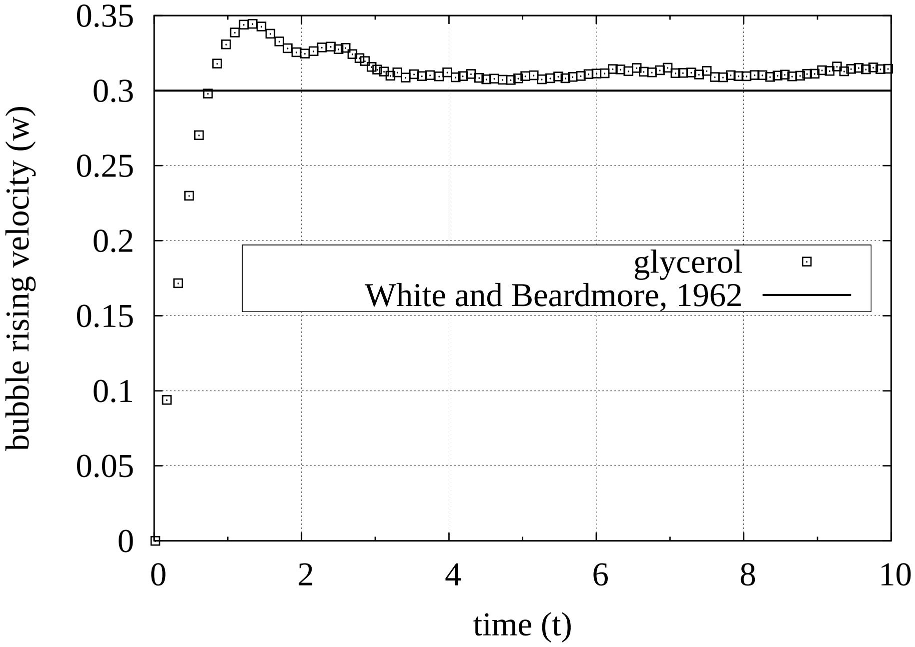
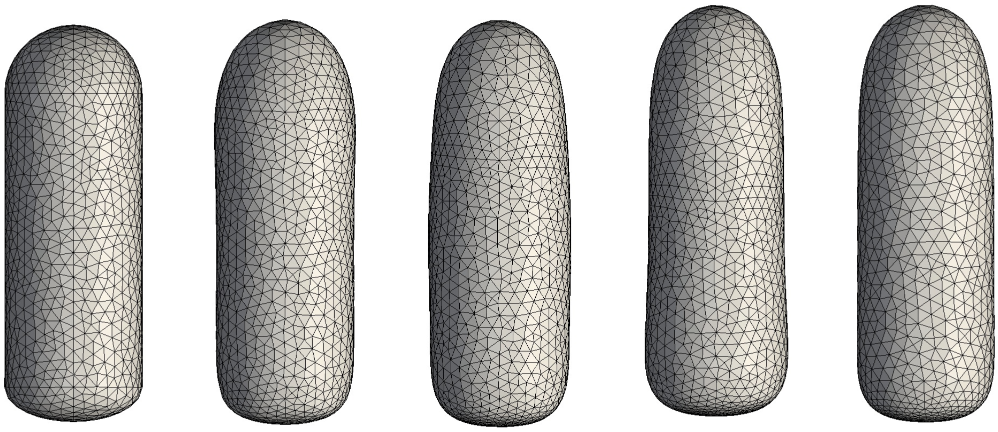


Figure 7.30: Rising of an air Taylor bubble immersed in a glycerol solution with dimensionless numbers set to $Mo = 10^{-2}$ and $Eo = 100$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional.



(a) $t = 0.00$ (b) $t = 0.70$ (c) $t = 1.93$ (d) $t = 3.35$ (e) $t = 8.76$

Figure 7.31: Bubble shape evolution with time for an air bubble in a glycerol solution with dimensionless numbers $Mo = 10^{-1}$, $Eo = 40$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 6.00$.

bubble and the wall. Moreover, the refined mesh close to the bubble assured the minimum number of elements located in the liquid film. Brown's correlation assumes that the liquid film

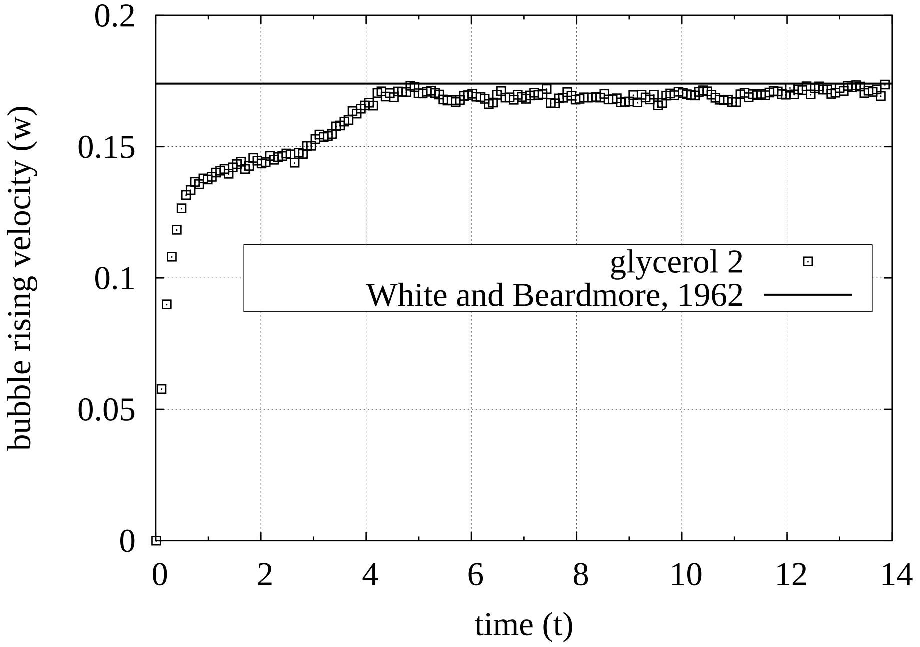


Figure 7.32: Rising of an air Taylor bubble immersed in a glycerol solution with dimensionless numbers set to $Mo = 10^{-2}$ and $Eo = 100$. The time evolution of the Bubble's center of mass velocity is compared to the terminal bubble's velocity found in [97]. Velocity and time are non-dimensional.

is stable, however during our simulations, the liquid film was not stable in all cases, suggesting that the large discrepancy of the computed film thickness for the solution of sucrose 1 may be the caused of this phenomena.

As an extension of the bubble's rising velocity and liquid film thickness, one fluid of Table (7.4) was selected to be compared to a 2-dimensional simulation. Thus, a quantitative and qualitative comparison can be done to estimate how far a 2-dimensional simulation compared to its 3-dimensional solution. The 2-dimensional code was implemented following the same guidelines for the 3-dimensional one. The equations are discretized by the Finite Element Method in the Arbitrary Lagrangian-Eulerian description, an adaptive mesh refinement is performed to keep the interface elements (triangle edges and nodes) and the triangles elements bounded to satisfactory shapes. The curvature is calculated using the 2-dimensional version of the *Frenet*'s formula (5.2) and the gravity is included in the y-direction, so that the surface tension and gravity effects are taken into account in the Navier-Stokes equation. Despite the reduced computational time used to perform this simulation, the moving frame technique was also applied to reproduce the same conditions of the 3-dimensional case. The numerical domain used to simulate this case was set to $D \times 5D + L_b$, where D stands for the channel diameter and L_b represents the bubble length, and thus the 2-dimensional domain is compatible to the 3-dimensional simulation. One test case was selected to be compared, and it is shown in Figs. (7.33 and 7.34). The *Morton* number was set to $Mo = 10E-02$ and the *Eötvös* number to

$Eo = 40$. Density ρ, viscosity μ and surface tension was set to those of the glycerol solution (see Table (7.4). The triangular mesh used comprised approximately 2445 nodes, 4430 triangles, 577 interface nodes (interface and boundary) and 577 edges to assemble the interface and boundary meshes. The bubble shape time evolution is shown in Fig. (7.33) and a qualitative comparison with the 3-dimensional simulations is shown in Fig. (7.34).

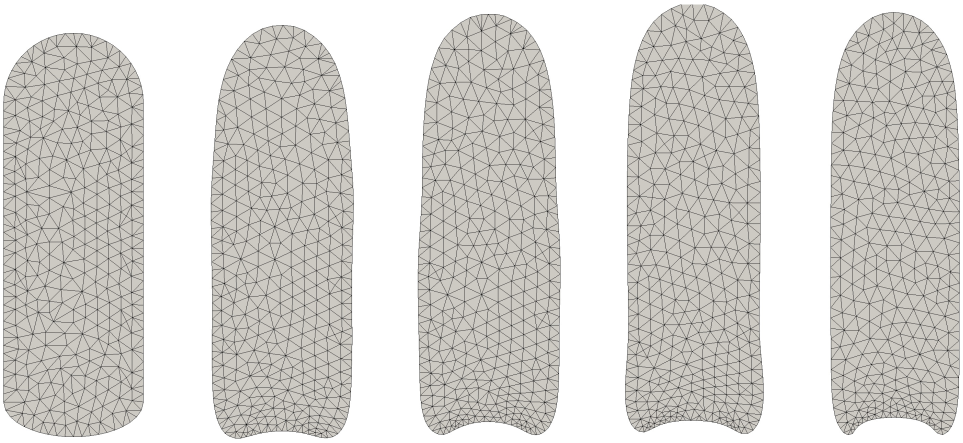

(a) $t = 0.00$ (b) $t = 1.14$ (c) $t = 1.63$ (d) $t = 2.54$ (e) $t = 7.62$

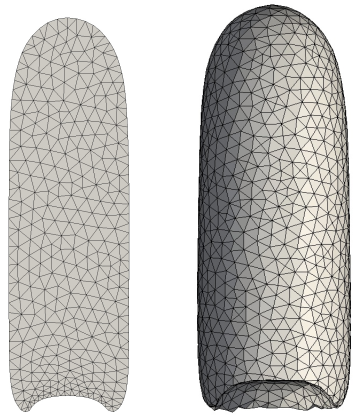
Figure 7.33: 2-dimensional bubble shape evolution with time for an air bubble in a glycerol solution with dimensionless numbers $Mo = 10^{-1}$, $Eo = 40$. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape during transient solution. (e) Terminal bubble shape at $t = 7.62$.

The results show that the terminal bubble shape is modeled relatively well in 2-dimensional simulations, which can be seen in Fig. (7.34). However the measured liquid film is not well predicted, being thicker compared to the 3-dimensional case, so thus the bubble length tends to be longer in 2-dimensions simulations. Another issue was found in the terminal rising velocity, which was found to be 25% than for the 3-dimensional case, therefore justifying the thicker liquid film.

It is important to note that the liquid film for the Taylor bubble simulations adds significantly the difficult of handling the mesh in the liquid film. However, with the proposed adaptive mesh refinement, the dynamics of the bubble rise could be essentially captured.

7.9 Two drop collision

The mechanisms of coalescence are still an open issue. A common approach is to assume a lubricating model to describe the thin liquid film dynamics of two approaching drops. In such a model, the interface between the fluids is considered to be two non deformable solid plates. However, the interface shape may modify during the coalescence process, thus if the curvature of the interface changes, the pressure field is immediately affected, and



(a) $t = 0.00$ (b) $t = 0.70$

Figure 7.34: Qualitative comparison of bubble's terminal shape in (a) 2-dimensional and (b) 3-dimensional simulations in a glycerol solution.

consequently the lubrication theory may not represent accurately the process. In [55], the mentioned analysis to study film drainage between a deformable interface was extended, improving the representation of the phenomena.

Coalescence has been modeled and described by breaking it down into three consecutive stages, namely approach of drops, film drainage and film rupture. The first stage is started when the drops are large and separated. The second stage is characterized by the approaching of the drops and the decreasing of the liquid film thickness between them. The last stage is represented by the rupture of the interface and consequent coalescence. An experimental work was performed by [101] and they pointed out the importance of the later stage, since the interface instabilities may be the cause of the film to rupture.

A numerical simulation of 3-dimensional drop collision was made by [66]. They investigated the coalescence process using a front-tracking code and they pointed out the limitation of their grid resolution to resolve the liquid film thickness between the drops. In [65], a numerical study was performed to investigate the collision of two droplets using an axisymmetric model. They compared their results to the available literature and obtained good agreement to the experimental results.

Taking advantage of the methodology developed in this work, where the nodes are moved in the Arbitrary Lagrangian-Eulerian description, a preliminary study was performed about the approaching of two equal-sized drops in general linear flows by considering two stagnation-points with constant strain conditions, namely planar and axisymmetric. In the former, the assumption of a planar flow is performed by imposing the boundary conditions according to the following expression:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{7.15}$$

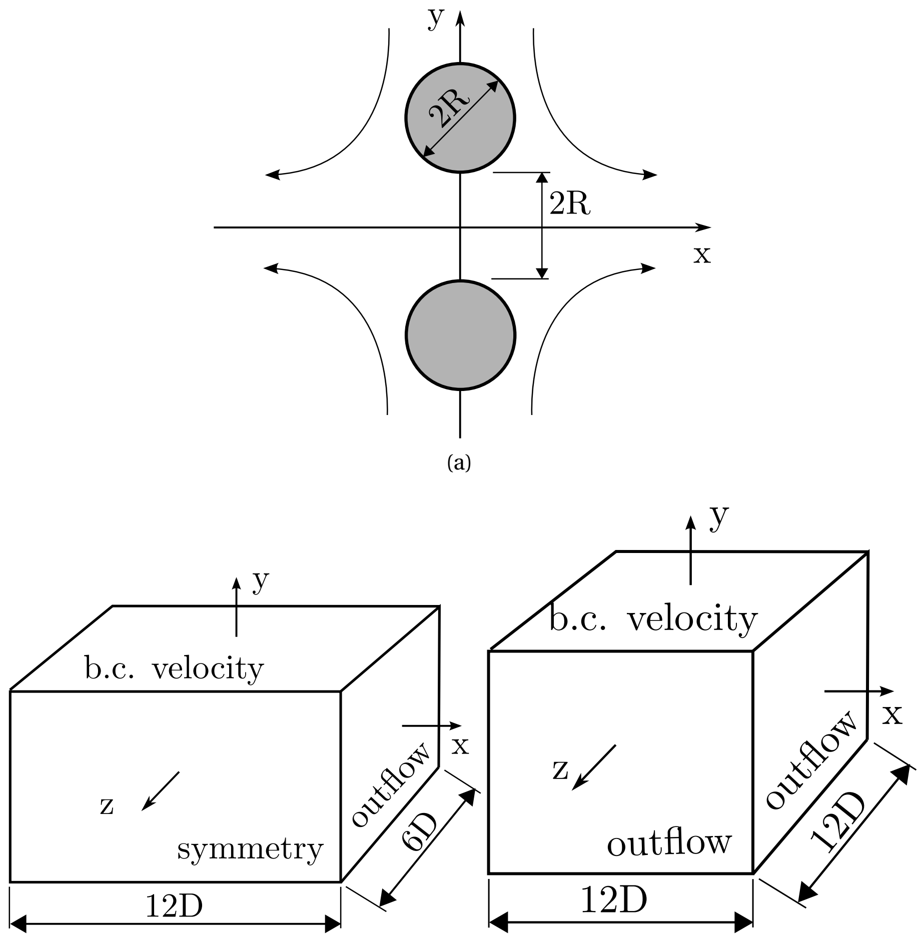
where u, v and w are the velocity components, and x, y and z are the boundary coordinates. Pressure is set in such a way that the flow is allowed to escape only in the perpendicular plane of the two drops. On the other hand, the boundary conditions imposed on the axisymmetric flows is slightly modified to take into account the w velocity component in the prescribed velocity, thus it is set according to the following expression:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{7.16}$$

The qualitative difference of these two boundary conditions is associated to the final bubble's shape and flow definition itself. In the planar case, the drop is squeezed along the $x-y$ plane and in the axisymmetric one, then the drop remains nearly spherical. In order to investigate the effects of the interface force during the approaching process of two equal-sized drops, the fluid properties were set equal to 1, such that $\rho_{in}/\rho_ontout = 1$, $\mu_{in}/\mu_ontout = 1$ and the *Weber* number is constant and set to $We = 2$. 2 cases were simulated for each flow condition by varying the *Reynolds* number $Re = \{2, 20, 200\}$, consequently varying the *Capillary* number $Ca\{1, 0.1, 0.01\}$. Such a condition is equivalent to the simulation of a soap bubble, where the inner and outer fluid are the same, but still the effect of an interface between the fluids is present.

Figure (7.35) depicts the background flow velocity and how the boundary conditions are applied in the two above mentioned cases. In the planar case, an outflow and a symmetry conditions are set in the perpendicular walls of the x and z-axis directions respectively. On the top $y = y_{max}$ and bottom $y = y_{min}$ parts, an inlet velocity condition is imposed. In the axisymmetric case, the difference in the imposition of the boundary condition is that the outflow is set at all the walls perpendicular to the x and z axis.

The shape evolution with time of the planar case is shown in Fig. (7.36). As can be seen, the higher is the Re number, the lower is the bubble's shape deformation along x direction. The thin liquid film between the two drops requires an extensive mesh controlling procedure to capture satisfactorily well the present mechanisms. Despite the planar background flow, the 3-dimensional simulation did capture a velocity gradient in the z direction, thus concluding that this flow can not be well predicted by a 2-dimensional code. However, a 2-dimensional


y
2R
2R
x
y
b.c. velocity
x
outflow
6D
z
symmetry
12D
y
b.c. velocity
x
z
outflow
outflow
12D
12D

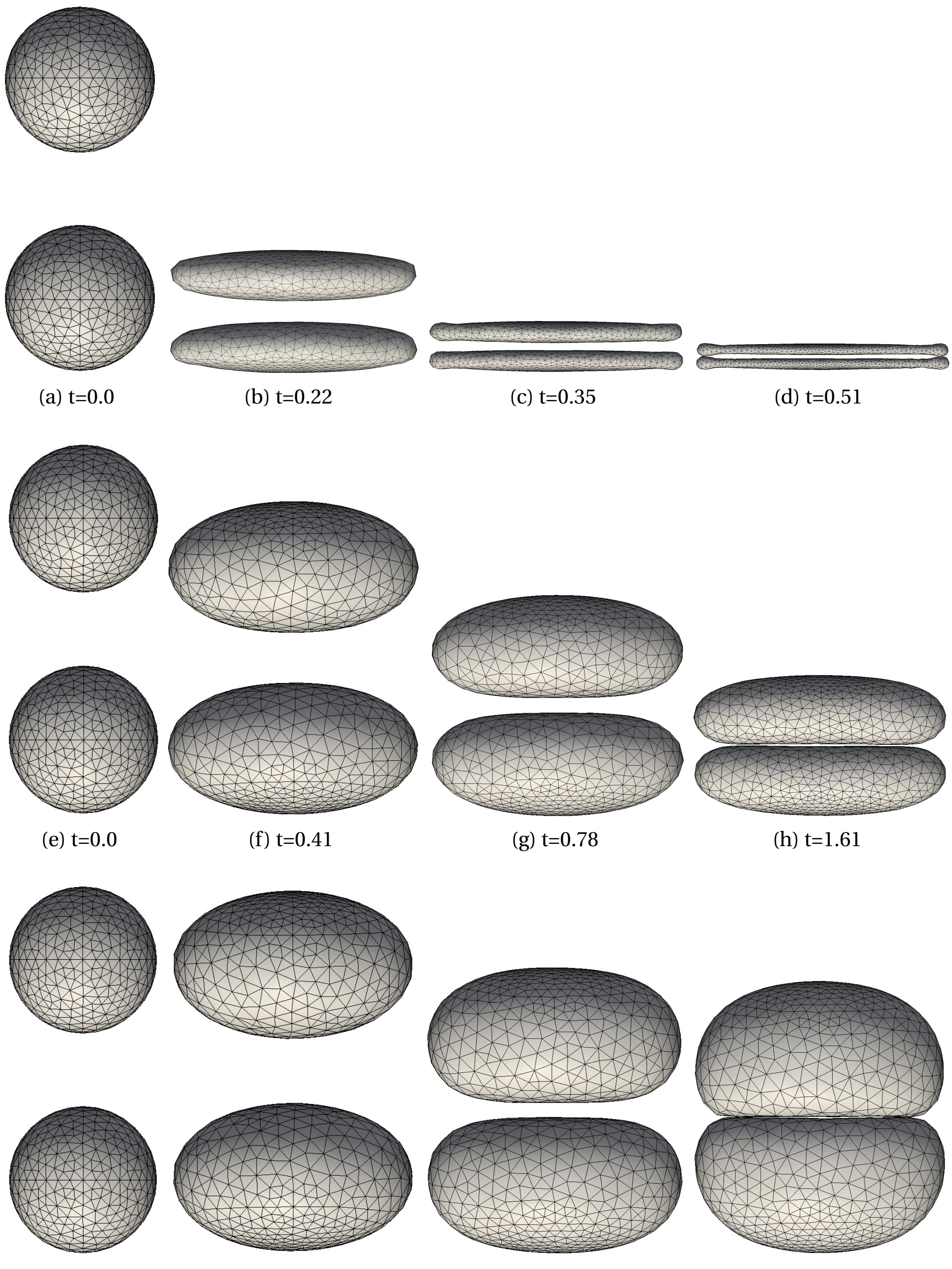
(a)

(b)

(c)

Figure 7.35: Boundary conditions (b.c.) applied to the two drop collision simulations. (a) Background flow characterized by stagnation-point with constant strain conditions. (b) Planar case. (c) Axisymmetric case.

simulation suggested that a qualitative description of this test case can be done.



(a) t=0.0 (b) t=0.22 (c) t=0.35 (d) t=0.51

(e) t=0.0 (f) t=0.41 (g) t=0.78 (h) t=1.61

(i) t=0.0 (j) t=0.28 (k) t=0.66 (l) t=0.93

Figure 7.36: Time evolution of the collision of two equal-sized drops in the planar flow imposed by Eq. (7.15). (a-d) Dimensionless parameters are $We = 2$ and $Re = 2$. (e-h) Dimensionless parameters are $We = 2$ and $Re = 20$. (i-l) Dimensionless parameters are $We = 2$ and $Re = 200$. Time is non-dimensional.

Figure (7.37) shows the time evolution of the collision of two equal-sized drops in the axisymmetric flow imposed by Eq. (7.16) for different Re numbers. In the axisymmetric case, the bubble's shape remains nearly spherical in the parallel plane, and thus it is squeezed in the perpendicular direction. It was observed different time scales for the approaching of the two simulated bubbles. This is due to the significant change in the Re number.

Figures (7.38a) and (7.38b) show the time evolution of the film thickness for the planar and axisymmetric cases respectively. As can be seen, if the Re number is high the trends tend to detach from the analytical solution of the flow field given by $0.5\mathrm{e}^{-x}$ and the linear evolution given by $0.5-0.5x$, where x is the coordinate, thus indicating that the non-linear effects are stronger in the liquid film.

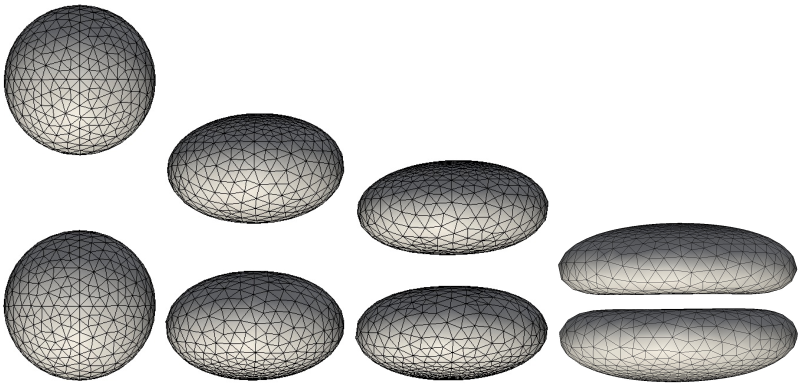
The computational mesh used at all simulations comprised approximately 11000 volumetric nodes, 66700 tetrahedrons, 1500 surface nodes and 2980 surface triangles. During the simulations, nodes are added and deleted, thus the numbers presented were not constant. The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.0$. The surface mesh parameter γ_2 was kept null to avoid excessive displacement of the surface mesh by the smoothing velocity.

In this preliminary study and based on the numerical simulations performed, it can be conclude that the flow can not be assumed to be linear in the gap between the two interfaces. Moreover, it has been seen that with the increasing of the Re number, the curvature of the interface in the liquid film region assumes negative values, thus the assumption of a non deformable surface is no longer valid. However, the effects of interface instabilities, pressure distribution and detailed velocity profiles still need to be further investigated. Furthermore, it appears that application of lubricating theory alone is not sufficient to model this process.

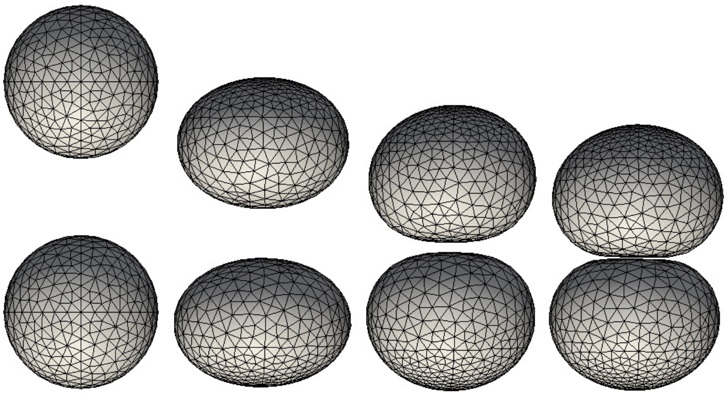
7.10 Microchannel simulations

Microchannel two-phase flows are important in many applications, in particular to slug flows in cooling of electronics as in the present CMOSAIC project. In such a scale, the surface tension force plays an important role in the flow dynamics, whereas gravitational effects are generally negligible. Thus, an accurate modelling of the surface tension force is required. Figure (7.39) depicts the test section of the microchannel boiling facility at the Heat and Mass Transfer Laboratory, where 67 microchannels with a square cross-section are distributed side-by side. Later, it will be placed on the top of the microprocessor, thus achieving highly efficient cooling by evaporating two-phase flow refrigerants.

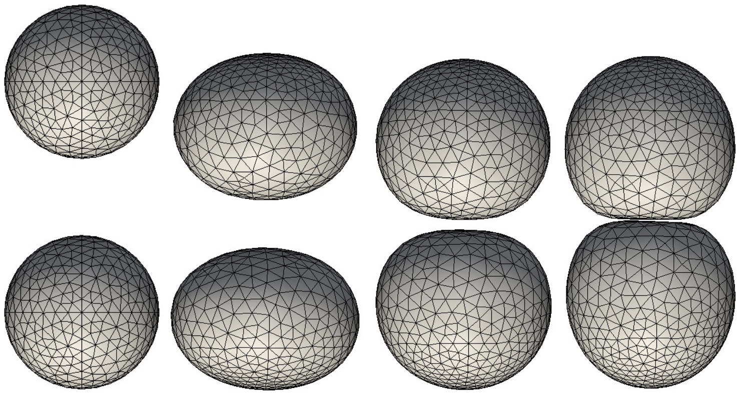
Four simulations were performed in the micro scale using different conditions and two fluids, namely R1234ze and R236fa, which are used in the scope of the cooling of 3D stacked chips in the CMOSAIC project. The properties of each working fluid can be seen in Table (7.6) and they are taken considering the working temperature of the refrigerants, which is 25°C. The length of the channel is on the order of 10^{-3}m. Figure (7.40) shows the surface mesh of the



(a) Re=2, t=0.0 (b) Re=2, t=0.67 (c) Re=2, t=1.29 (d) Re=2, t=2.85



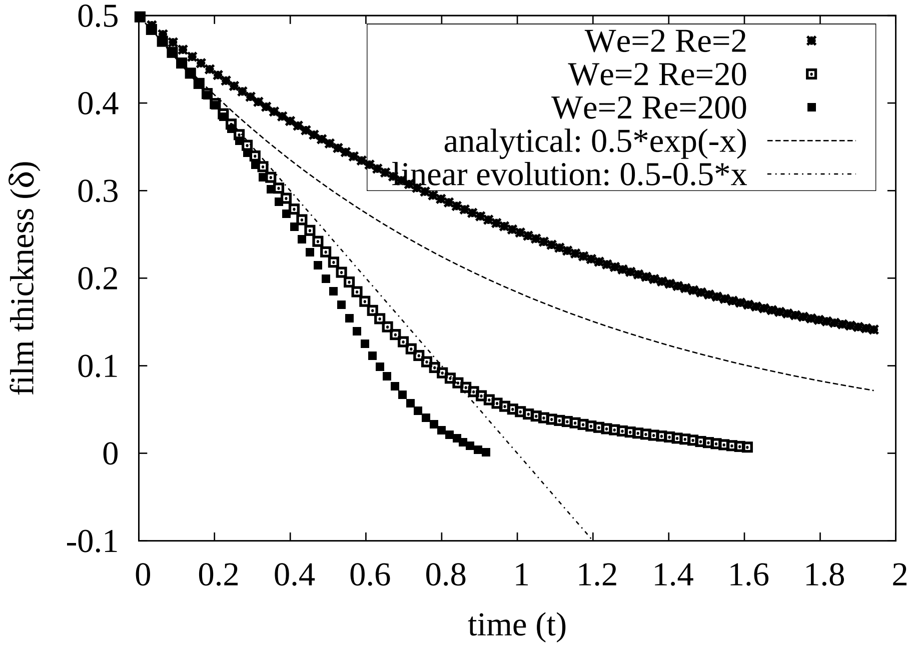
(e) Re=20, t=0.0 (f) Re=20, t=0.45 (g) Re=20, t=1.09 (h) Re=20, t=1.74



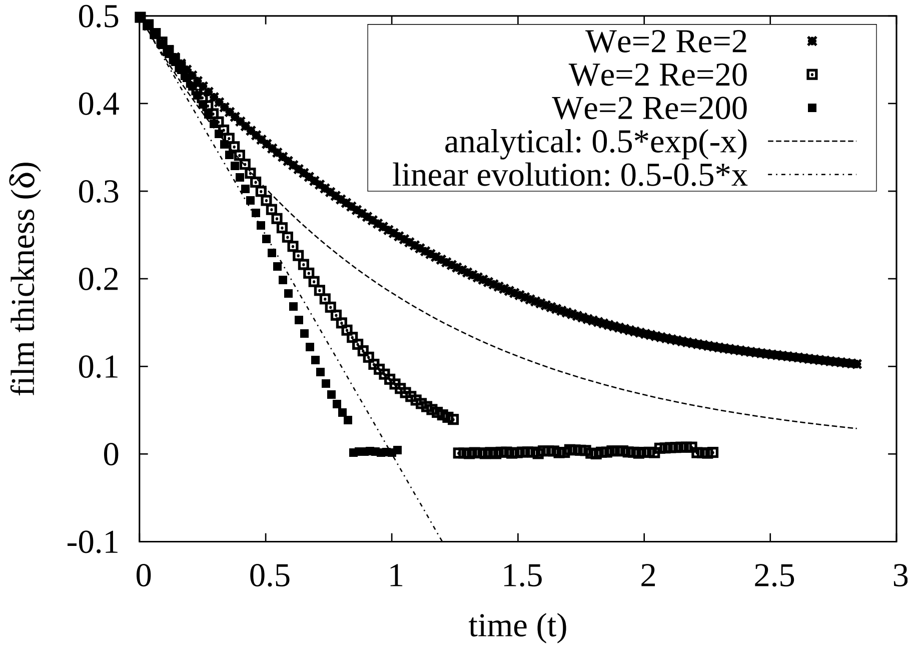
(i) Re=200, t=0.0 (j) Re=200, t=0.45 (k) Re=200, t=0.78 (l) Re=200, t=0.93

Figure 7.37: Time evolution of the collision of two equal-sized drops in the axisymmetric flow imposed by Eq. (7.16). (a-d) Dimensionless parameters are $We = 2$ and $Re = 2$. (e-h) Dimensionless parameters are $We = 2$ and $Re = 20$. (i-l) Dimensionless parameters are $We = 2$ and $Re = 200$. Time is non-dimensional.

3-dimensional domain used to simulate a single bubble flowing inside a micro channel with a square cross-section of 100x100μm. The bubble is placed near the middle of the domain,



(a)



(b)

Figure 7.38: Film thickness δ of (a) planar case, where the z component of the velocity is not considered on the boundary conditions and (b) axisymmetric case

and the mesh receives a special treatment to capture the thin liquid film found in such flows. Therefore, in order to simulate such a condition, the same idea of the moving referential frame

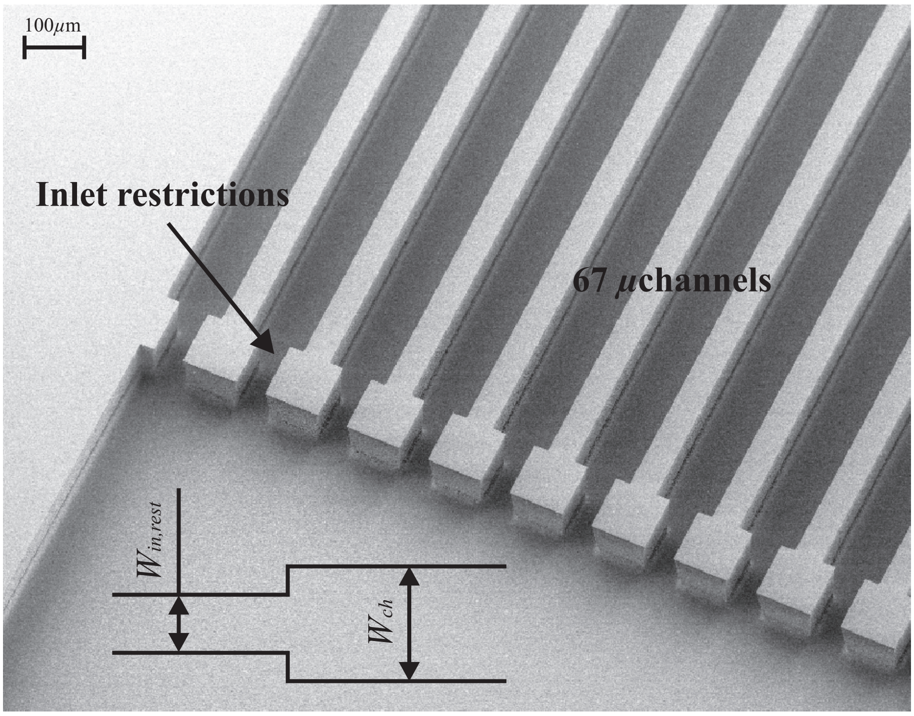


Figure 7.39: Test section of the microchannel boiling facility at the Heat and Mass Transfer Laboratory. 67 microchannes with square cross-section are distributed side-by-side and placed on the top of the micro processor. The square channel length is 100μm. Figure extracted from [57].

is used to shorten the numerical domain, and the boundary conditions are applied to include the wall effect into the bubble dynamics. Since the flow is driven by the shear effect and the boundaries will move with the bubble's velocity, a velocity profile was set on the left and right walls. On the remaining walls, the no slip condition was used and a single node is used to set the pressure condition, thus satisfying the continuity equation in Eq. (3.13). Such an imposition resembles the lid-driven cavity, in which only one single node is required to solve the pressure field.

The isothermal simulation of an isolated bubble is performed to evaluate the conditions found in microchannel flows using refrigerant R236fa as the working fluid. The bubble's length was set to $L_b = 1.2D$ where D is the cross-section width. Figure (7.41) shows the evolution of the bubble's shape with time and colored by the velocity component in the x-direction. After a transient stage, the bubble reaches its stable solution for the bubble shape. However, the tail of the bubble presents a small continuous oscillation, typically observed experimentally. The dimensionless liquid film thickness remained stable and its minimum value was measured, and it was found to be $\delta_{min} = 0.057$, which represents 57μm.

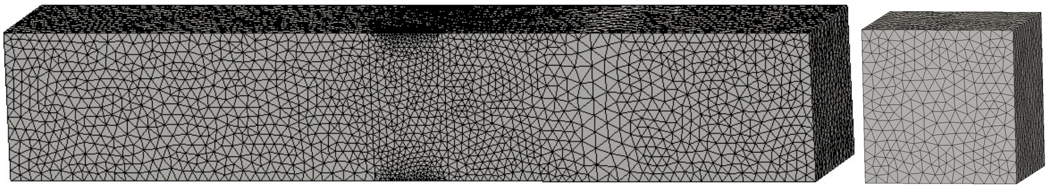

(a) (b)

Figure 7.40: Numerical domain used to simulate the microchannel boiling facility's microchannel. (a) The side view of the domain shows the mesh distribution, where a denser cluster of nodes are located near the bubble region. (b) The square microchannel cross-section.

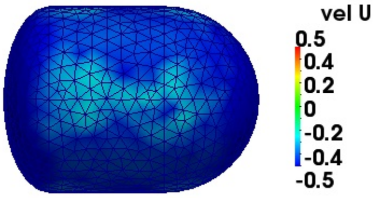
Table 7.6: Fluid properties

fluid	vapor phase				liquid phase				interface
	ρ $[kg/m^3]$	μ $[\mu Pa\cdot s]$	k $[mW/mK]$	c_p $[kJ/kgK]$	ρ $[kg/m^3]$	μ $[\mu Pa\cdot s]$	k $[mW/mK]$	c_p $[kJ/kgK]$	σ $[mN/m]$
R1234ze	25.898	12.464	15.664	1.0421	1157.4	202.71	62.060	1.3882	8.5156
R236fa	18.294	10.846	12.760	0.8844	1360.3	286.03	72.870	1.2641	10.086

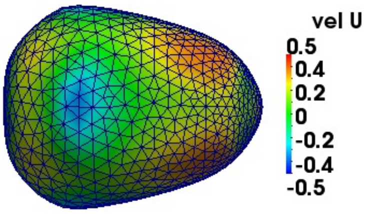
A different working fluid was tested with a longer bubble length $L_b = 2D$, where D is the width of the square channel cross-section. Figure (7.42) depicts the time evolution of a single vapor bubble of the refrigerant R1234ze. As can be seen, the bubble changes its shape from time $t = 0.0$ to $t \approx 5$, the liquid film becomes thinner and surface waves appear close to the tail of the vapor bubble, which also continuously oscillates, as observed in experiments. The colors represent the velocity of the surface nodes in the x-direction.

A more interesting test case was simulated using the same refrigerant as in the previous test case, but with an additional bubble placed $0.5D$ behind the other one. Therefore, the second bubble is affected by the wake formed by the motion of the first one, characterizing a slug flow. Each bubble length was set to be $L_b = 2D$. Figure (7.43) depicts the bubble's shape evolution of two vapor bubbles of the refrigerant R1234fa. Different patterns were found in the bubble's shapes as well as the evolution of the film thickness with time, as can be seen in Fig. (7.44). The liquid film thicknesses were found to be $\delta = 20\mu m$ and $\delta = 37\mu m$ for the left and right bubble respectively. From time $t \approx 8$ till the end of the simulation, the two bubbles remained equally distanced.

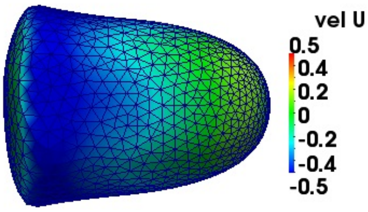
The dynamics of a single vapor bubble of the refrigerant R1234ze was simulated taking into account the evaporation process produced by the constant and uniform heat flux $\dot{q}$ applied to the bottom part (wall) of the numerical domain. Thus, the simulation of the microchannel placed on the top of a working chip can be achieved. The bubble's length was set to $L_b = 1.3D$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

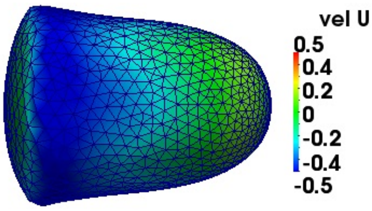
(a) $t = 0.00$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

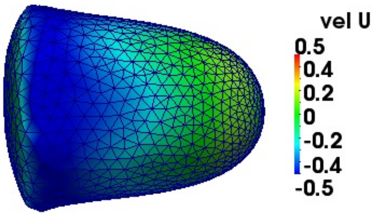
(b) $t = 0.70$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(c) $t = 1.93$

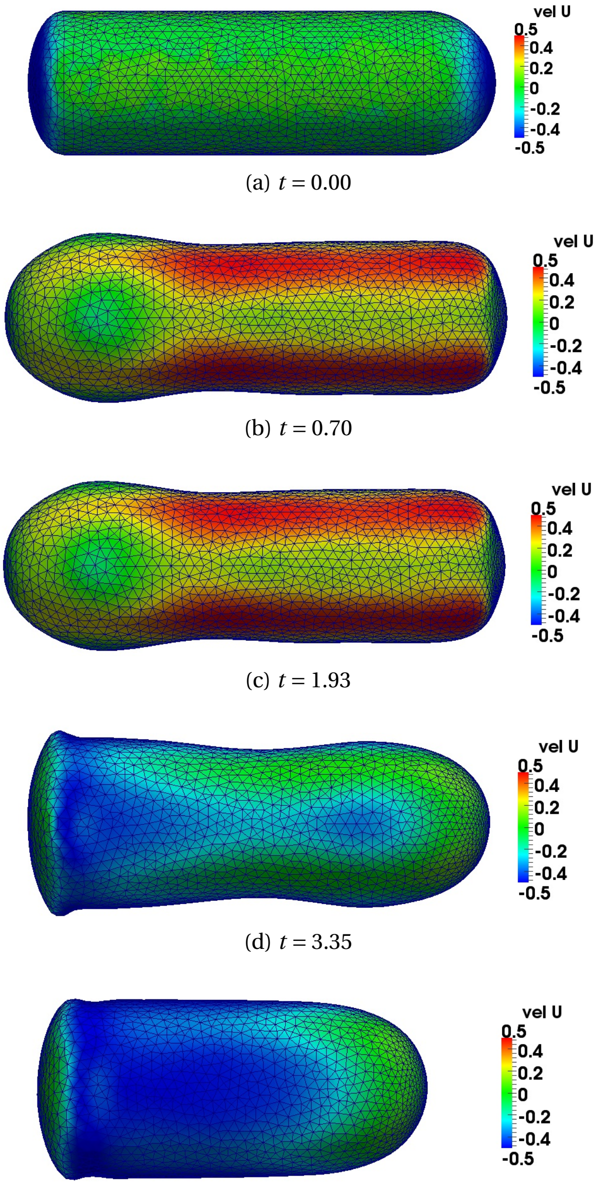

vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(d) $t = 3.35$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(e) $t = 8.76$

Figure 7.41: Bubble's shape evolution with time for a single vapor bubble. The working fluid is R236fa, whose details can be found in the Table (7.6) The fluid is entering at the left of the domain and exiting at the right. A transient stage occurs in (a-b), followed by a nearly stable bubble shape in (c-e).


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5
vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5
vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5
vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5
vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(a) $t = 0.00$

(b) $t = 0.70$

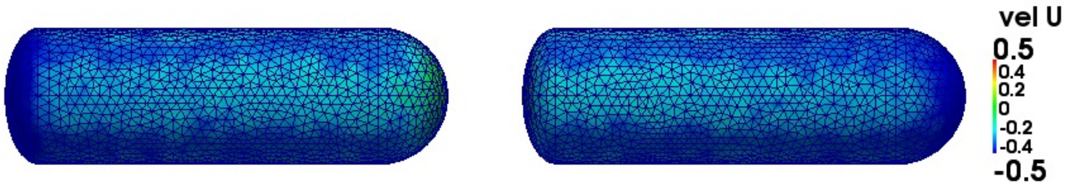
(c) $t = 1.93$

(d) $t = 3.35$

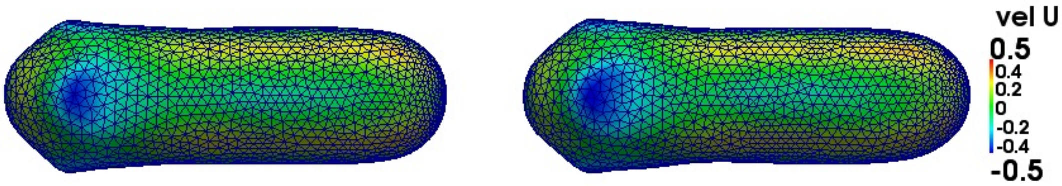
(e) $t = 8.76$

Figure 7.42: Bubble's shape evolution with time for a single vapor bubble. The working fluid is the recent environmentally friendly refrigerant R1234ze, whose details can be found in Table (7.6). (a-c) Transient stage. (d) initial formation of the surface waves in the tail of the vapor bubble. (d) Stable bubble shape.

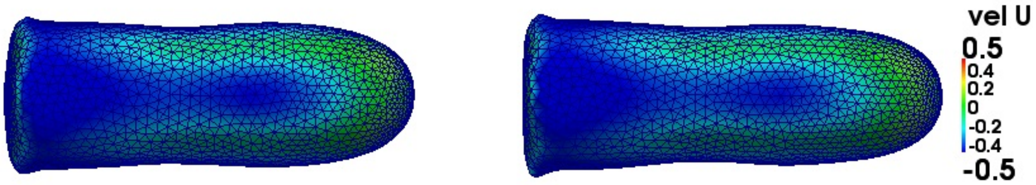
and the initial film thickness to $\delta_i = 0.35$. Figure (7.45) shows the transient solution of the temperature profile of two-phase evaporation of R1234ze. Due to the evaporation process, it was verified that the bubble's volume changed 12% compared to the initial volume. Moreover, the transient bubble's shape presented a different pattern compared to the isothermal simulation (Fig. (7.42), indicating that the evaporation of the liquid film affected the bubble's shape.


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

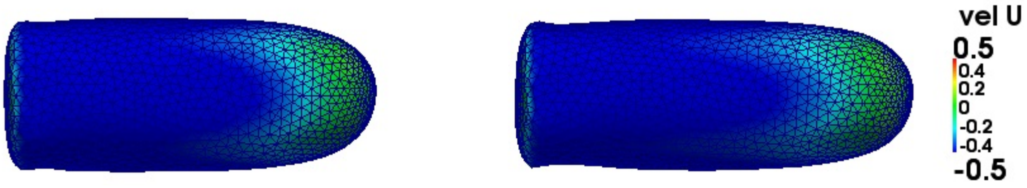
(a) $t = 0.00$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

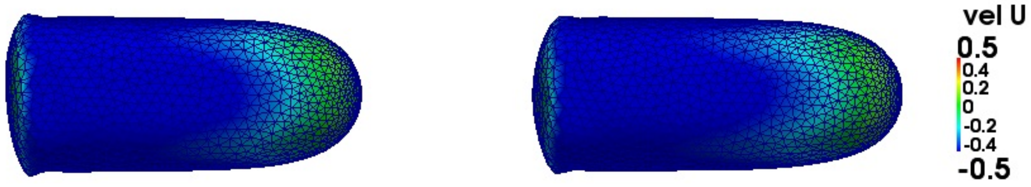
(b) $t = 0.70$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(c) $t = 1.93$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(d) $t = 3.35$


vel U
0.5
0.4
0.2
0
-0.2
-0.4
-0.5

(e) $t = 8.76$

Figure 7.43: Bubble's shape evolution with time for two vapor bubbles. The working fluid is refrigerant R1234ze, whose details can be found in Table (7.6). The fluid is entering at the left of the domain and exiting at the right. (a-c) Transient stage. (d) initial formation of the surface waves in the tail of the vapor bubble. (d) Stable bubble shape.

The surface waves were not identified in the tail of the bubble. The measured film thickness found was $\delta = 0.073$. A further investigation is still required to validate the implemented mass transfer model.

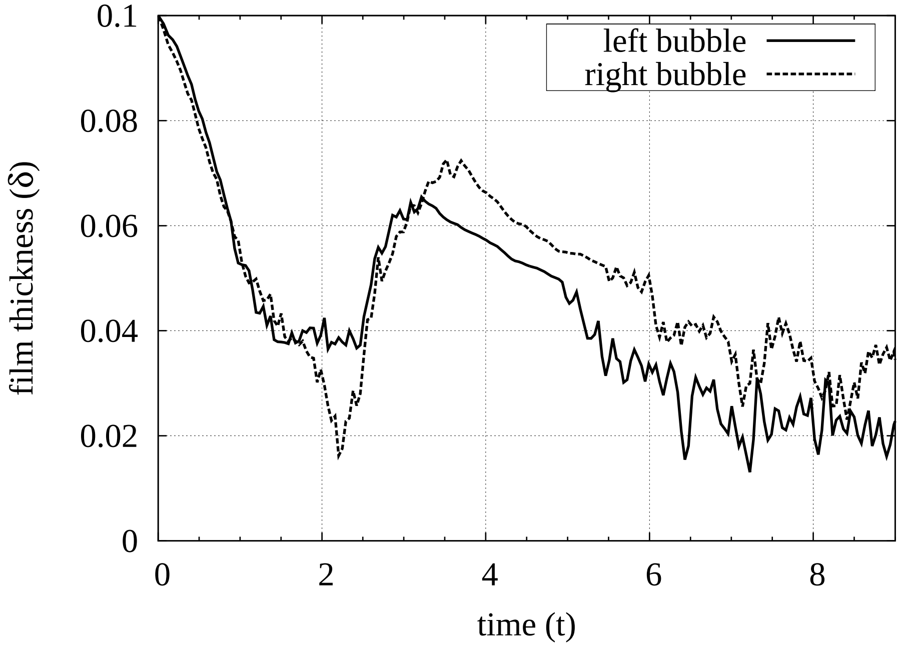
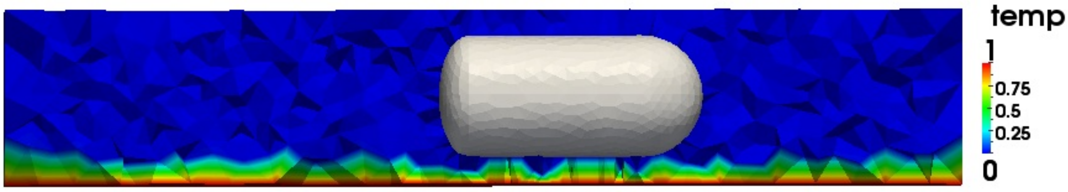


Figure 7.44: Transient solution of the liquid film thickness computed for the two vapor bubbles of the refrigerant R1234ze.

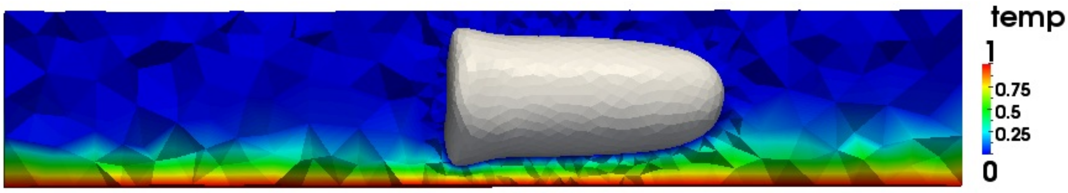
Finally, a comparison is made between 2-dimensional and 3-dimensional simulations with the same conditions of the single vapor bubble of refrigerant R1234ze presented previously. The triangular mesh had approximately the same number of nodes and elements of the above mentioned 2-dimensional case. The bubble shape evolution with time is shown in Fig. (7.46) and a qualitative comparison with the 3-dimensional simulations is shown in Fig. (7.47).

The results show that the terminal bubble shape is not well modeled in the 2-dimensional simulation, which can also be seen in Fig. (7.34). Moreover, the measured liquid film is not well predicted, being higher if compared to the 3-dimensional case, and thus the bubble length tends to be longer in the 2-dimensions simulation. Another issue was found in the terminal velocity, being 21% than for the 3-dimensional case. Therefore, it can be conclude that a 3-dimensional simulation is required to simulate the dynamics found in the present test cases.

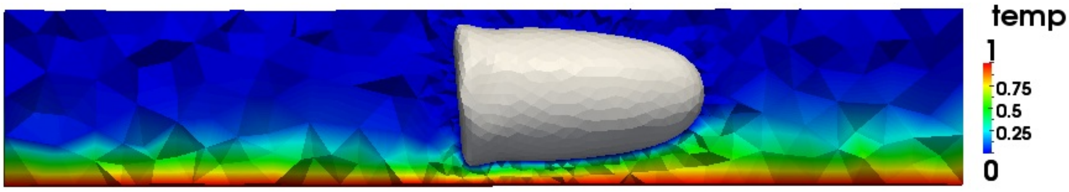
The computational mesh used in the single bubble microchannel simulations comprised approximately 13000 volumetric nodes, 60000 tetrahedrons, 5500 surface nodes and 11000 surface triangles. In the two bubbles test case, the computational mesh comprised 22000 volumetric nodes, 140000 tetrahedrons, 11000 surface nodes and 22000 surface triangles. During the simulations, nodes are added and deleted, thus the numbers presented were not constant. The mesh parameters used in this simulation were $\beta_1 = 0.0$, $\beta_2 = 1.0$, $\gamma_1 = 1.0$ and $\gamma_2 = 0.0$.


temp
1
0.75
0.5
0.25
0

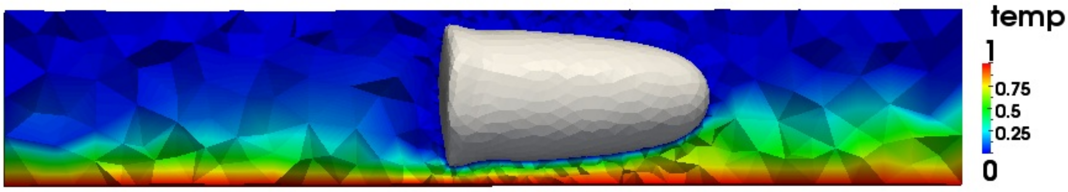
(a) $t = 0.00$


temp
1
0.75
0.5
0.25
0

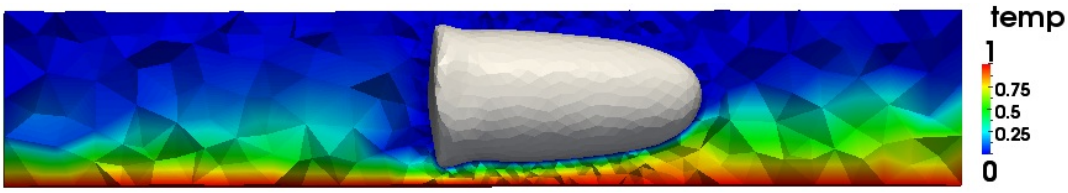
(b) $t = 3.03$


temp
1
0.75
0.5
0.25
0

(c) $t = 4.52$

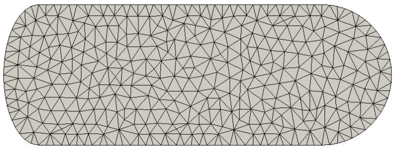

temp
1
0.75
0.5
0.25
0

(d) $t = 8.41$

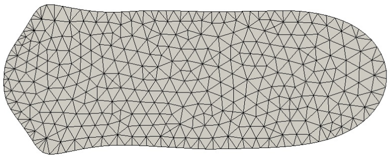

temp
1
0.75
0.5
0.25
0

(e) $t = 13.03$

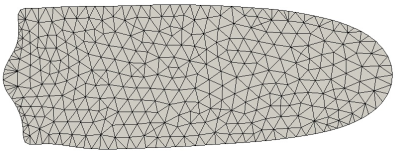
Figure 7.45: Transient solution of two-phase flow boiling of refrigerant R1234ze. A constant heat flux $\dot{q}$ is applied in the bottom part of the domain. The fluid is entering at the left of the domain and exiting at the right. Time and temperature are non-dimensional.



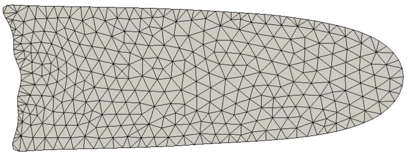
(a) $t = 0.00$



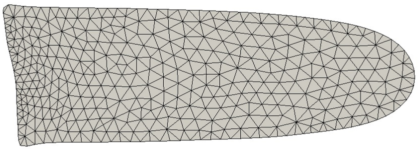
(b) $t = 0.91$



(c) $t = 2.30$

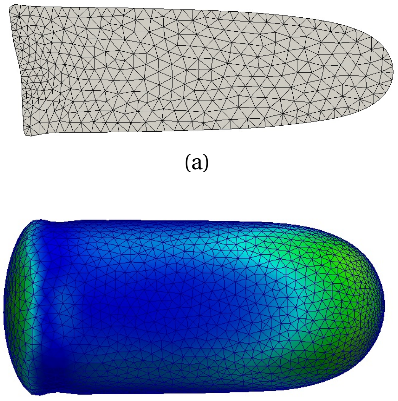


(d) $t = 4.26$



(e) $t = 7.60$

Figure 7.46: 2-dimensional bubble shape evolution with time for a vapor bubble of refrigerant R1234ze. (a) Initial bubble shape with $t = 0$. (b-d) Bubble shape transient solution. (e) Terminal bubble shape with $t = 7.62$.



(a)

(b)

Figure 7.47: Qualitative comparison of bubble’s terminal shape in (a) 2-dimensional and (b) 3-dimensional simulations of refrigerant R1234ze.

8 Conclusions

This thesis presents a new methodology for simulating incompressible two-phase flows within the Finite Element Method context in which the mesh moves in an Arbitrary Lagrangian-Eulerian fashion. The coupling FEM-ALE methodology provides a sharp representation of the interface between the phases, not only for the geometrical representation itself but also for the definition of the phase properties, thus resulting in a model which accurately describes the actual physical conditions.

The new methodology proposed for the calculation of the surface tension force, based on the *Frenet*'s formula, has been shown to be consistent and accurate with moderate programming effort and computational cost. An exclusive test was performed to compare the curvature calculation for different geometries where the two principal curvatures may not necessarily present equal signs. The results have shown to be bounded to the maximum relative error of 5% of the analytical curvature.

The proposed treatment of the computational mesh, splitting the surface meshes and the volumetric points, has shown to be an excellent choice, thus avoiding the obstacles of handling the remeshing process over the tetrahedron mesh, allowing the utilization of a standard Delaunay tetrahedralization library. Moreover, the new adaptive meshing strategy achieves good control of the mesh quality during the simulations by keeping the volumetric and surface elements bounded to a satisfactory shape, thus preserving the accuracy of the calculation. However, excessive linear interpolations on the volumetric and surface mesh may lead to a lack of accuracy in certain domain areas and this should be used with care.

Benchmarks and results for the static droplet, sessile drop, oscillating drop and falling drop in an inert media have also shown good agreement with their respective analytical solutions. Additionally, the rising bubble tests were successfully compared to an experimental database found in the classical literature for different mesh refinement levels. Thus, these tests highlight the proposed ALE-FEM scheme's suitability to use as an accurate and adaptable two-phase flow simulation strategy. Moreover, the code has shown to be suitable to capture the dynamics of the liquid film and to model systems with multiple bubbles and drops as shown in the two drop collision and the microchannel simulations. Furthermore, it was shown that an equivalent 2-d simulation was significantly different for the simulation of the 3-d elongated bubble in a microchannel, demonstrating that the present 3-d code is needed while a 2-d code is not sufficient.

The heat and mass transfer was implemented into the code using the same strategy of the fluid flow solver. However, it requires an extensive reformulation of the isothermal method presented here. The thin thermal boundary layer requires an additional number of nodes and elements to be physically resolved, thus increasing significantly the processing time. The preliminary tests presented for evaporation in square microchannels qualitativly agreed to those observed experimentally. However, a deeper analysis is still required to compare the actual state of the phase change model with different benchmarks available in the literature.

In summary, a new two-phase flow simulation methodology has been proposed and proven to stand up to numerous benchmarks and tests and is now available for simulations in microchannel flows.

8.1 Further work

Within the development of the proposed method, many difficulties were sucessfuly overcome and new ideas were brought into the numerical code. However, various improvements can still be implemented to enhence the code's capability, thus allowing the modeling of more complex phenomena. Taking advantage of the flexibility of the code's design, it is proposed that further developments should address:

- parallelization: enable high performance computing schemes to increase the processing performance of the simulations, thus reducing the input time;
- coalescence: modeling of the process by which two or more bubbles are merged during contact.
- static and dynamic contact angle: interactions of bubbles or drops to rigid walls, allowing the modeling of more complex phenomena;
- surface roughness of microchannels: different roughness partterns can be tested and included as boundary conditions in order to approach the physical reality;
- sub-grid models: thermal micro layer and bubble break-up are examples of sub-grid models that could enhence the code's capabilities.

A Important theorems

Theorem 1 [Integration by parts] Be $\Omega \subset \mathbb{R}^m$, $\Gamma = \partial\Omega$ the boundary of Ω and $\phi, \psi : U \subset \mathbb{R}^m \to \mathbb{R}$ scalar fields, therefore:

$$\int_\Omega \phi \nabla \psi \cdot \mathbf{n} \, d\Omega = \int_\Gamma \phi \psi \, d\Gamma - \int_\Omega \psi \nabla \phi \cdot \mathbf{n} \, d\Omega$$

Theorem 2 [The first form of Green]: Be $\Omega \subset \mathbb{R}^m$, $\Gamma = \partial\Omega$ the boundary of Ω and $\phi, \psi : U \subset \mathbb{R}^m \to \mathbb{R}$ scalar fields, therefore:

$$\int_\Omega \left(\phi \nabla^2 \psi + \nabla \phi \cdot \nabla \psi \right) \, d\Omega = \int_\Gamma \phi \nabla \psi \cdot \mathbf{n} \, d\Gamma \, .$$

Theorem 3 [Second form of Green]: Be $\Omega \subset \mathbb{R}^m$, $\Gamma = \partial\Omega$ the boundary of Ω and $\phi, \psi : U \subset \mathbb{R}^m \to \mathbb{R}$ scalar fields, therefore:

$$\int_\Omega \left(\phi \nabla^2 \psi + \psi \nabla^2 \phi \right) \, d\Omega = \int_\Gamma \left(\phi \frac{\partial \psi}{\partial n} + \psi \frac{\partial \phi}{\partial n} \right) \, d\Gamma \, .$$

Theorem 4 [Green's theorem for vector fields]: Be $\Omega \subset \mathbb{R}^m$, $\Gamma = \partial\Omega$ the boundary of Ω and $\mathbf{u}, \mathbf{w} : U \subset \mathbb{R}^m \to \mathbb{R}^m$ vector fields, therefore:

$$\int_\Omega \left\{ (\nabla^2 \mathbf{u}) \cdot \mathbf{w} + (\nabla \mathbf{u} : \nabla \mathbf{w}^{\mathbf{T}}) \right\} \, d\Omega = \int_\Gamma \mathbf{n} \cdot (\nabla \mathbf{u} \cdot \mathbf{w}) \, d\Gamma \, .$$

Theorem 5 [Gauss's divergence]: Be $\Omega \subset \mathbb{R}^m$, $\Gamma = \partial\Omega$ the boundary of Ω and $\mathbf{u} : U \subset \mathbb{R}^m \to \mathbb{R}^m$

a vector field, therefore:

$$\int_{\Omega} \nabla \cdot \mathbf{u} \, d\Omega = \int_{\Gamma} \mathbf{u} \cdot \mathbf{n} \, d\Gamma \, .$$

where **n** is the normal vector in Γ.

B TetGen - command line switches

The command syntax used to generate different meshes are presented below:

- -p Tetrahedralizes a picecwise linear complex (.poly or .smesh file).
- -q Quality mesh generation. A minimum radius-edge ratio may be specifyed (default 2.0).
- -a Applies a maximum tetrahedron volume constraint.
- -A Assigns attributes to identify tetrahedra in certain regions.
- -r Reconstructs/Refines a previously generated mesh.
- -Y Suppresses boundary facets/segments splitting.
- -i Inserts a list of additional points into mesh.
- -M Does not merge coplanar facets.
- -T Set a tolerance for coplanar test (default 1e-8).
- -d Detect intersections of PLC facets.
- -z Numbers all output items starting from zero.
- -j Jettison unused vertices from output .node file
- -o2 Generates second-order subparametric elements.
- -f Outputs faces (including non-boundary faces) to .face file.
- -e Outputs subsegments to .edge file.
- -n Outputs tetrahedra neighbors to .neigh file.
- -g Outputs mesh to .mesh file for viewing by Medit.

- -G Outputs mesh to .msh file for viewing by Gid.
- -O Outputs mesh to .off file for viewing by Geomview.
- -B Suppresses output of boundary information.
- -N Suppresses output of .node file.
- -E Suppresses output of .ele file.
- -F Suppresses output of .face file.
- -I Suppresses mesh iteration numbers.
- -C Checks the consistency of the final mesh.
- -Q Quiet: No terminal output except errors.
- -V Verbose: Detailed information on what I'm doing.
- -v Prints the version information.
- -h Help: A brief instruction for using TetGen.

At all simulations, the following command was invoked in the numerical code:

- `tetrahedralize( (char*) "QYYApa",&in,&out );`

where &in,&out are the surface and volumetric meshes respectively.

Bibliography

[1] Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engeneering*, 194:4135–4195, 2004.

[2] M. Ainsworth and B. Senior. Aspects of an adaptive @finite element method: Adaptive strategy. *Computer Methods in Applied Mechanics and Engeneering*, 150:65–87, 1997.

[3] A. Amsden and F H. Harlow. The smac method: A numerical technique for calculating incompressible fluid flows. Technical report, Los Alamos, 1970.

[4] G.R. Anjos. Hydrodynamics field solution of electrochemical cells through finite element method. Master's thesis, Metallurgical and Materials Engineering, Federal University of Rio de Janeiro, Brazil, 2007.

[5] G.R. Anjos, N. Mangiavacchi, J. Pontes, and C.P. Botelho. Modelagem numérica de escoamentos acoplados ao transporte de uma espécie química pelo método dos elementos finitos. ENCIT 2006 - Congresso Brasileiro de Ciências Térmicas e Engenharia, 2006. Curitiba - PR.

[6] G.R. Anjos, N. Mangiavacchi, J. Pontes, and C.P. Botelho. Simulaćão numérica das equaćões de saint-venant utilizando o método dos elementos finitos. 16 POSMEC - Simpósio de Pós-Graduaćão em Engenharia Mecânica, 2006. Uberlândia - MG.

[7] S. Balay, J. Brown, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.2, Argonne National Laboratory, 2011.

[8] S. Balay, W.D. Gropp, L.C. McInnes, and B.F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[9] C.E. Baumann and J.T. Oden. A discontinuous hp finite element method for convection-diffusion problems. *Computer Methods in Applied Mechanics Engeneering*, 175:311–341, 1998.

[10] Y. Bazilevs, L.B. Veiga, J.A. Cottrell, T.J.R. Hughes, and G. Sangalli. Isogeometric analysis: approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences*, 7:1031–1090, 2006.

[11] D. Bhaga and M.E. Weber. Bubbles in viscous liquids: Shapes, wakes and velocities. *Journal of Fluid Mechanics*, 105:61–85, 1981.

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

[13] H. Braess and P. Wriggers. Arbitrary lagrangian eulerian finite element analysis of free surface flows. *Computer methods in applied mechanics and engineering*, 190:95–109, 2000.

[14] R.A.S Brown. The mechanism of large bubbles intubes. i. bubbles velocities in stagnant liquids. *Canadian Journal of Chemical Engineering*, 43:217–223, 1965.

[15] J.D Bugg, K. Mack, and K. S. Rezkallah. A numerical model of taylor bubbles rising through stagnant liquids in vertical tubes. *International Journal of Multiphase Flow*, 24:271–281, 1998.

[16] W. Chang, F. Giraldo, and B. Perot. Analysis of an exact fractional step method. *Journal of Computational Physics*, 2002.

[17] L. Chen and Y. Li. A numerical method for two phase flows with an interface. *Enviromental Modeling and Software*, 13:247–255, 1998.

[18] S.-W Cheng and T.K. Dey. Maintaining deforming surface meshes. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 112–121, 2008.

[19] S.-W Cheng and J. Jin. Edge flips and deforming surface meshes. In *Proceedings of the 27th Annual Symposium on Computational Geometry*, 2011.

[20] A. J. Chorin. Numerical solution of the navier-stokes equations. *Mathematics of Computation*, 22:745–762, 1968.

[21] C. Cuvelier, A. Segal, and A. A. van Steenhoven. *Finite Element Method and Navier-Stokes Equations.* Dordrecht, Holland, 1986.

[22] M. Desbrum, M. Meyer, P. Schrder, and A. Barr. Implicit fairing of arbitrary meshes using diffusion and curvature flow. In *Proceedings of Siggraph*, pages 317–324, 1999.

[23] O. Devillers. On deletion in delaunay triangulations. *International Journal of Computational Geometry & Applications*, 12:285, 2002.

[24] K.D. Devine and J.E. Flaherty. Parallel adaptive hp-refinement techniques for conservation laws. *Applied Numerical Mathematic*, 20:367–386, 1996.

[25] V.K. Dhir. Nucleate and transition boiling heat transfer under pool and external flow conditions. *International Journal of Heat and Fluid Flow*, 21:290–314, 1991.

[26] J. Donea. A taylor-galerkin method for convective transport problems. *Int. J. Num. Methods Eng.*, 20:101–19, 1984.

[27] D.R. Durran. *Numerical Methods for Waves Equations in Geophysical Fluid Dynamics*. Springer-Verlag, 1a. edition, 1998.

[28] H. Edelsbrunner, X.-Y. Li, G.L. Miller, A. Stathopoulos, D. Talmor, S.-H. Teng, A. Üngör, and N. Walkington. Smoothing cleans up sliver. In *Proceedings of 32nd Annual ACM Symposium in Theory of Computation*, pages 273–277, 2000.

[29] A. Esmaeeli and G. Tryggvason. Computations of explosive boiling in microgravity. *Journal of Scientific Computating*, 19:163–182, 2003.

[30] A.O. Fortuna. *Técnicas Computacionais para Dinâmica dos Fluidos*. edUSP, 1a. edition, 2000.

[31] M.M. Francois, S.J. Cummins, E.D. Dendy, D.B. Kothe, J.M. Sicilian, and M.W. Williams. A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework. *Journal of Computational Physics*, 213:141–173, 2006.

[32] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[33] I. Ginzburg and G. Wittum. Two-phase flows on interface refined grids modeled with vof, staggered finite volumes, and spline interpolants. *Journal of Computational Physics*, 166:302–335, 2000.

[34] J. Glimm, J. Grove, W.B. Lindquist, O. McBryan, and G. Tryggvadson. The bifurcation of tracked scalar waves. *SIAM Journal of Computations*, 9(1):61–79, 1988.

[35] J.M. González-Yuste, R. Montenegro, Escobar J.M., G. Montero, and E. Rodríguez. Local refinement of 3-d triangulations using object-oriented methods. *Advances in Engineering Software*, 35:693–702, 2004.

[36] W.J. Gordon and L.C. Thiel. Numerical grid generation, 1982. Edited by J.F. Thompson.

[37] J.R. Grace. Shapes and velocities of bubbles rising in infinite liquids. *Transactions of the Institution of Chemical Engineering*, 51:116–120, 1973.

[38] P. Gresho. On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via the finite element method that also introduces a nearly consistent mass matrix. part1: Theory. *International Journal for Numerical Methods in Fluids*, 11:587–620, 1990.

[39] F H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, 8:2182–2189, 1965.

[40] C.W. Hirt and B.D. Nichols. Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39:201–225, 1981.

[41] J. Hua, J.F. Stene, and P. Lin. Numerical simulation of 3d bubbles rising in viscous liquids using a front tracking method. *Journal of Computational Physics*, 227(6):3358–3382, 2008.

[42] T. Hughes, L. Franca, and M. Balestra. A new finite element formulation for computational fluid dynamics: V. circunventing the babuska-brezzi condition: A stable petrov-galerkin formulation of the stokes problem accomodating equal-order interpolations. *Computer Methodos in Applied Mechanics and Engineering*, 59:85–99, 1985.

[43] T. Hughes, L. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: I. symmetric forms of the compressible euler and navier-stokes equation and the second law of thermodynamics. *Computer Methodos in Applied Mechanics and Engineering*, 54:223–34, 1986.

[44] T. J. R. Hughes. *The Finite Element Method - Linear Static and Dynamic finite Element Analysis.* Dover Publications, INC, 1987.

[45] T.J.R. Hughes. Lagrangian-eulerian finite element formulatin for incompressible viscous flows. *Computational Methdos Applied to Mechancal Engineering*, 29:329–349, 1981.

[46] V. Jayaraman, H.S. Udaykumar, and W.S. Shyy. Adaptive unstructured grid for three-dimensional interface representation. *Numerical Heat Transfer, Part B: Fundamentals*, 32:247–265, 1997.

[47] D. Juric and G. Tryggvason. Computations of boiling flows. *Journal of Computational Physics*, 24(3):387–410, 1998.

[48] E. Kreyszig. *Differential Geometry.* Dover, New York, 1a. edition, 1991.

[49] T.N. Krishnamurti. Numerical integration of primitive equations by a quasi-lagrangian advective scheme. *Journal of applied Meteorology*, 1:508–521, 1962.

[50] C. kunkelmann and P. Stephan. Cfd simulation of boiling flows using the volume-of-fluid method within openfoam. *Numerical Heat Transfer, Part A: Applications*, 56(8):631–646, 2009.

[51] 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*, 2001.

[52] R. W. Lewis, P. Nithiarasu, and K. N. Seetharamu. *Fundamentals of the Finite Element Method for Heat and Fluid Flow.* Wiley John and Sons, 2004.

[53] R. Löhner, K. Morgan, and O.C. Zienkiewicz. An adaptive finite element procedure for compressible high speed flows. *Computer Methodos in Applied Mechanics and Engineering*, 51:441–65, 1985.

[54] M.A M.A. Walkley, P.H. Gaskell, P.K. Jimack, M.A. Kelmanson, and J.L. Summers. Finite element simulation of three-dimensional free-surface flow problems with dynamic contact lines. *International journal for numerical methods in fluids*, 00:1–6, 2004.

[55] G. D. M. MacKay and S. G. Mason. The gravity approach and coalescence of fluid drops at liquid interfaces. *The Canadian Journal of Chemical Engineering*, 41:203–212, 1963.

[56] E. Marchandise and J-F. Remacle. A numerical method for the simulation of three-dimensional incompressible two-phase flows is presented. *Journal of Computational Physics*, 219:780–800, 2006.

[57] M.M. Mohamed M. Sabry, A. Sridhar, D. Atienza, Y. Temiz, Y. Leblebici, S. Szczukiewicz, N. Borhani, J.R. Thome, T. Brunschwiler, and B. Michel. Towards thermally-aware design of 3d mpsocs with inter-tier cooling. In *Proceedings of Design, Automation and Test in Europe (DATE)*, pages 1466–1471, 2011. ACM and IEEE Press.

[58] A. Mukherjee and V.K. Dhir. Study of lateral merger of vapor bubbles during nucleate pool boiling. *Journal of Heat Transfer*, 126:1023–1039, 2004.

[59] S. Nagrath, K.E. Jansen, and R.T. Lahey. Computation of incompressible bubble dynamics with a stabilized finite element level set method. *Computer Methods in Applied Mechanics and Engineering*, 194:4565–4587, 2005.

[60] S. Nebuloni. *Numerical Modeling of Annular Laminar Film Condensation in Circular and Non-Circular Micro-Channels under Normal and Micro-Gravity*. PhD thesis, EPFL - Swiss Federal Institute of Technology, 2010.

[61] S. Nebuloni and J.R. Thome. Numerical modeling of laminar annular film condensation for different channel shapes. 53(13-14):2615–2627, 2010.

[62] S. Negami. Diagonal flips in triangulation of surfaces. *Discrete Mathematics*, (135):225–232, 1994.

[63] B.A. Nichita. *An Improved CFD Toll to Simulate Adiabatic and Diabatic Two-Phase Flows*. PhD thesis, EPFL - Swiss Federal Institute of Technology, 2010.

[64] B.A. Nichita, I Zun, and J.R. Thome. A level set method coupled with a volume of fluid method for modeling of gas-liquid interface in bubbly flow. *Journal of Fluids Engineering*, 132(8):081302, 2010.

[65] N. Nikolopoulos, K.-S. Nikas, and G. Bergeles. A numerical investigation of central binary collision of droplets. *Computer and Fluids*, 38:1191–1202, 2009.

[66] M.R. Nobari and G. Tryggvason. Numerical simulation of three dimensional droplet collision. *AIAA Journal*, 34(4):750–5, 1996.

[67] J. T. Oden and G.F. Carey. *Finite Elements: Mathematical Aspects.* Prentice-Hall, vol. iv edition, 1984.

[68] J.T. Oden, I. Babuska, and C.E. Baumann. A discontinuous hp finite element method for diffusion problems. *Journal of Computational Physics.*, 146:491–519, 1998.

[69] A. Okabe, B. Boots, K. Sugihara, and S. Nok Chiu. *Spatial Tessellations - Concepts and Applications of Voronoi Diagrams.* John Wiley and Sons, 2000.

[70] S. Osher and J.A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[71] S. V. Patankar. *Numerical heat transfer and fluid flow.* Hemisphere, 1980.

[72] 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, 2002.

[73] O. Pironneau. On the transport-diffusion algorithm and its applications to the navier-stokes equation. *Numerische Mathematik*, 38:309–332, 1982.

[74] J. Pontes, N. Mangiavacchi, and G.R. Anjos. *Estabilidade Hidrodinâmica em Células Eletroquímicas - Modelagem Computacional em Materiais.* Editora Ciência Moderna, 1 edition, 2010.

[75] R.J. Purser and L.M. Leslie. Generalized adams-bashforth time integration schemes for a semi-lagrangian model employing the second-derivative of the horizontal momentum equations. *Quarterly Journal of the Royal Meteorological Society*, 122:737–763, 1996.

[76] S. Quan and D.P. Schmidt. A moving mesh interface tracking method for 3d incompressible two-phase flows. *Journal of Computational Physics*, 221:761–780, 2006.

[77] M. Quecedo and M. Pastor. Application of the level set method to the finite element solution of two phase flows. *International Journal for Numerical Methods in Engineering*, 50:645–663, 2001.

[78] A. Robert. A stable numerical integration scheme for the primitive meteorological equations. *Atmosphere Oceans*, 19:35–46, 1981.

[79] J.S. Sawyer. A semi-lagrangian method of solving the vorticity advection equation. *Tellus*, 15:336–342, 1963.

[80] L.E. Scriven. On the dynamics of phase growth. *Chemical Engineering Science*, 10:1–13, 1959.

[81] J. R. Shewchuk. *Delaunay Refinement Mesh Generation.* PhD thesis, School of Computer Science, Pittsburg, USA, 1997.

[82] H. Si. Adaptive tetrahedral mesh generation by constrained delaunay refinement. *International Journal for Numerical Methods in Engineering*, 46(7):856–880, 2008.

[83] G. Son. A numerical method for bubble motion with phase change. *Numerical Heat Transfer, Part B*, 39:500–523, 2001.

[84] F.S Souza and N. Mangiavacchi. A lagrangian level-set approach for the simulation of incompressible two-fluid flows. *International Journal for Numerical Methods in Fluids*, 47:1393–1401, 2004.

[85] M. Sussman. A second order coupled level-set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *Journal of Computational Physics*, 187:110–136, 2003.

[86] M. Sussman and P. Smereka. Axisymmetric free boundary problems. *Journal of Fluid Mechanics*, 341:269–294, 1997.

[87] M. Sussman, P. Smereka, and S. Osher. A level-set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.

[88] G. Taubin. Geometric signal processing on polygonal meshs. In *Proceedings of Eurographics*, pages 1–11, 2000.

[89] T.E. Tezduyar. Finite element methods for flow problems with moving boundaries and interfaces. *Archives of Computational Methods in Engineering*, 8(2):83–130, 2001.

[90] A. Tomiyama. Struggle with computational bubble dynamics. In *Proceedings of Third Conference on Multiphase Flow - ICMF*, June 8-12 1998.

[91] A. Tomiyama, I. Zun, H. Higaki, Y. Makino, and T. Sakaguchi. A three-dimensional particle tracking method for bubbly flow simulation. *Nuclear Engineering and Design*, 175:77–86.

[92] A. Tomiyama, I. Zun, A. Sou, and T. Sakaguchi. Numerical analysis of bubble motion with the vof method. *Nuclear Engineering and Design*, 141:69–82, 1993.

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

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

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

[96] G. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. *Journal für die Reine und Angewandte Mathematik*, 1908.

[97] E.T. White and R.H. Beardmore. The velocity of rise of single cylindrical air bubbles through liquids contained in vertical tubes. *Chemical Engineering Science*, 17:351–361, 1962.

[98] A. Wiin-Nielsen. On the application of trajectory methods in numerical forecasting. *Tellus*, 11:180–196, 1959.

[99] J. Wu, ST. Yu, and BN Jiang. Simulation of two-fluid flows by the least-squares finite element method using a continuum surface tension model. *International Journal for Numerical Methods in Engineering*, 42:583–600, 1998.

[100] X. Xu, C.C. Pain, A.J.H Goddard, and C.R.E. Oliveira. An automatic adaptive meshing technique for delaunay triangulations. *Computer Methods in Applied Mechanics and Engineering*, 161:297–303, 1998.

[101] H. Yang, C.C. Park, Y.T Hu, and L.G. Leal. The coalescence of two equal-sized drops in a two-dimensional linear flow. *International Journal of Multiphase Flow*, 13(5):1087–1106, 2001.

[102] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method for Fluids Dynamics*. Wiley John and Sons, 5th edition edition, 2000.

[103] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method Volume 1: The Basis*. Wiley John and Sons, 5th edition edition, 2000.

[104] O.C. Zienkiewicz and Y.K. Cheung. *Finite elements in the solution of field problems*. The Engineer, 507-10 edition, 1965.

Gustavo Rabello dos Anjos

Professional address: Bureau: ME G0 520
EPFL - STI - IGM - LTCM
ME Station 9
CH-1015 Lausanne, Switzerland
Office number: +41 21 6935442
E-mail: gustavo.rabello@epfl.ch
Webpage: http://ltcm.epfl.ch
Personal webpage: http://gustavo.rabello.org

Home address: Av. Druey 32
CH-1018 Lausanne, Switzerland
Phone number: +41 21 6473817
Mobile number: +41 762328894
E-mail: gustavo.rabello@gmail.com
Born on September 5th,1980 in Rio de Janeiro
Civil state: married

FORMATION

2008-2012 École Polytechnique Fédérale de Lausanne (EPFL)
@: http://www.epfl.ch
Ph.D. at Heat and Mass Transfer Laboratory
Director: John R. THOME and Navid Borhani
Thesis: **A 3D ALE Finite Element Method for Two-Phase Flows with Phase Change**

2005-2007 Federal University of Rio de Janeiro, (UFRJ / COPPE)
@: http://www.ufrj.br
Masters at Metallurgy and Materials Engineering Department
Director: Jose PONTES and Norberto MANGIAVACCHI
Thesis: **Hydrodynamic Field Solution on Electrochemical Cells Through Finite Element Method**

2000-2005 State University of Rio de Janeiro (UERJ)
@: http://www.uerj.br
Bachelor at Mechanical Engineering Department
Director: Mila R. AVELINO
Project: **Atmospheric Boundary Layer Simulation on Wind Tunnel**

LANGUAGES

- portuguese - mother language
- english - read, write and speak
- french - read, write and speak
- spanish - read and listen (elementary level)

INFORMATICS

Numerical languages

C/C++, python, fortran, matlab, bash

Operating systems

Unix/Linux, MacOS and Windows network administration

PROFESSIONAL EXPERIENCE

2007-2008 GESAR - Group of Environmental Simulations of Hydropower Reservoirs
State University of Rio de Janeiro, UERJ
Mechanical Engineering Department

Abstract: software development of a numerical simulator capable of predicting the dynamical properties of hydropower reservoirs fulfillment. The discretization of Navier-Stokes's equations was made by the Finite Element Method.

PUBLICATIONS & CONGRESSES

1. ANJOS, G.R., Borhani, N., Mangiavacchi, N., Thome, J.R. - A 3D ALE-FEM Method for Two-Phase Flows - **Journal of Computational Physics**, 2012 (submitted).

2. ANJOS, G.R., Mangiavacchi, N., Pontes, J, Mattos, O.R. - Rotating Disk Flow in Electrochemical Cells: A Three-Dimensional Finite Element Method Formulation, **Journal of Electrochemical Society**, 2012 (submitted).

3. ANJOS, G.R., Borhani, N., Mangiavacchi, N., Thome, J.R. - A 3D ALE- Finite Element Method for Two-Phase Flows with Phase Change. Lausanne, Switzerland **8th International Conference on Boiling and Condensation**, 2012.

4. ANJOS, G.R., Borhani, Thome, J.R. - A 3D ALE-FEM Simulation for Two-Phase Flows with Phase Change, Udine, Italy **50th European Two-Phase Flow Group Meeting**, 2012.

5. ANJOS, G.R., Borhani, N., Mangiavacchi, N., Thome, J.R. - 3D Moving Mesh Technique for Microscale Two-Phase Flows, Tel-Aviv, Israel, **49th European Two-Phase Flow Group Meeting**, 2011.

6. ANJOS, G.R., Borhani, N., Thome, J.R. - A 3D ALE-FEM Method for Microscale Two-Phase Flows, London, USA, **48th European Two-Phase Flow Group Meeting**, 2010.

WORKSHOP & SEMINARS

1. A 3D ALE-FEM Simulation of Microscale Two-Phase Flows **CFD Developments**, EPFL, Lausanne, Switzerland, 12th. March 2012.

2. 3D Moving mesh simulation for Microscale Two-Phase Flows, **Workshop Energy and Environment**, UERJ – State University of Rio de Janeiro, Rio de Janeiro, Brazil, 4th – 6th July 2011.

3. 3D ALE-FEM Microscale Two-Phase Flows, **3rd. Computational Fluid Dynamic Workshop**, UERJ - State University of Rio de Janeiro, Rio de Janeiro, Brazil, 3-9th. May 2011.

4. Finite Element Method applied to Two-Phase Flows. **Two-Phase Flow and Heat Transfer Numerical Workshop**, EPFL, Lausanne, Switzerland, 31st March 2011.

5. Numerical Simulation of Microscale Two-Phase Flows - An Arbitrary Lagrangian Eulerian Approach, Two-Phase Flow **Dynamics and Heat Transfer Workshop**, EPFL, Lausanne, Switzerland, 15th February 2011.

6. Finite Element and the Surface Tension Model, **1st. Computational Fluid Dynamic Workshop**, UERJ - State University of Rio de Janeiro, Rio de Janeiro, Brazil, 6th. May 2009.