POLITECNICO
MILANO 1863

Department of Aerospace Science and Technology
Doctoral Program in Aerospace Engineering

An adaptive interpolation-free conservative scheme for the three-dimensional Euler equations on dynamic meshes for aeronautical applications

PhD candidate: **Barbara Re**
Mat. 803083

Supervisor:
Prof. Alberto Guardone

Co–supervisor:
Prof. Cécile Dobrzynski

Tutor:
Prof. Luigi Vigevano

The Chair of the Doctoral Program:
Prof. Luigi Vigevano

XXVIII cycle — March 2016

A Marco

Acknowledgments

At the end of this work, I would like to thank all people who supported and helped me during these years. Having people at my side who trusted me was essential to persist and attain the (long-awaited) results.

First of all, I am very grateful to my supervisor, Prof. Guardone, for the possibility of carrying out this research and for his constant help and support. Without his precious suggestions, most of the obstacles that I encountered during the work would not have been overcome. I enjoyed working with him and I would like to thank him also for the time spent talking ‘about doing research’ and the future prospects. I want to express my thankfulness to Prof. Dobrzynski, who welcomed me at Inria-Bordeaux and taught me almost all I know about grids and mesh adaptation. Without her help, the remaining part of the problems would have remained unsolved. Then, I am grateful to Prof. Dolejsi for the time he spent on reviewing this manuscript and for providing valuable suggestions.

Moreover, I would like to thank all colleagues and friends (in Milan and Bordeaux) that made the time spent working on this thesis more pleasant. A big thank also to all friends who prevented me from taking myself and this work too seriously. Finally, a special thank goes to my family, in particular to mom and dad, for their unreserved support and for their comprehension of my bad moon due to work problems, especially when the results were long in coming.

At last, I would like to say a big thank you to Marco. Your encouragement helped me more than you can imagine.

Abstract

A novel approach to solve the finite volume formulation for unsteady inviscid compressible flows over three-dimensional dynamic grids is proposed. The major novelty of the proposed approach concerns the interpretation of local mesh modifications—node displacement, deletion, addition or edge swap—by means of a three-steps procedure that allows to compute the volume change due to mesh adaptation as series of fictitious continuous deformations of the finite volumes. Thanks to this procedure, the solution on the new grid is recovered by exploiting the Arbitrary Lagrangian-Eulerian formulation of the Euler equation without any explicit interpolation, even if grid connectivity changes occur. This allows to preserve the scheme properties, especially conservativeness and monotonicity, and to implement multi-step time schemes in a straightforward way. More precisely, conservativeness is enforced by an appropriate computation of the grid velocity that allows to automatically fulfill the Geometric Conservation Law.

The robust strategy adopted in the present work consists in moving the mesh keeping the topology fixed until the quality falls below a certain threshold, then local mesh adaptation is performed to re-store mesh quality and prevent element entanglement. Furthermore, mesh adaptation is exploited to increase solution accuracy, according to a suitable target grid spacing, which in unsteady simulations is obtained from the solution computed in the prediction step carried out after the mesh deformation. Indeed, no delay between mesh adaptation and the actual geometry is introduced.

The validity of the proposed conservative adaptive interpolation-free strategy is first assessed by different simulations of reference test cases, in which a good agreement with the reference results have been obtained, then three-dimensional simulations of aeronautical interest are carried out. Mesh adaptation is successfully exploited in the steady simulation of the transonic flow around the ONERA M6 wing to capture the peculiar lambda-shock that forms over the upper wing surface. The capability of the proposed approach to accurately capture the relevant flow features is assessed also in the unsteady simulations of a pitching infinite-span NACA 0012 wing, where the bi-dimensional character of the flow is well reproduced and a good agreement with the bi-dimensional results obtained for the pitching airfoil is observed. Finally, the capability of dealing with large boundary displacement is assessed by computing the flow around the infinite-span NACA 0012 wing in the laboratory reference frame, namely a quiescent flow is enforced over the boundary domain and the wing travels through the domain at the flight velocity. With respect to the standard steady simulation in the wing reference frame, the results of the unsteady simulation show a lim-

ited oscillation around the steady value. Analogous results are obtained also in a similar simulation over a finite-span wing, assessing that the proposed strategy is well-suited to perform accurate conservative simulations of moving-body problems over adaptive grids.

Keywords:

Arbitrary Lagrangian-Eulerian scheme, Mesh adaptation, Large boundary deformations, Euler equations.

Contents

Acknowledgments **I**

Abstract **III**

1 Introduction **1**
1.1 Motivation and goal . . . 1
1.2 Numerical solution of the flow equations on moving boundaries . . . 2
1.2.1 Arbitrary Lagrangian-Eulerian framework . . . 2
1.2.2 Small boundary displacement: dynamic grids . . . 4
1.2.3 GCL: an essential constraint for deforming meshing . . . 5
1.2.4 Large boundary displacement . . . 5
1.3 Mesh adaptation techniques . . . 7
1.3.1 Adaptation criteria . . . 8
1.3.2 Adaptation strategy . . . 9
1.3.3 Unsteady mesh adaptation . . . 10
1.4 Thesis goals and outline . . . 10

2 Finite Volume ALE scheme for three-dimensional flows **13**
2.1 Euler equations . . . 13
2.1.1 Thermodynamic model . . . 14
2.1.2 Non-dimensional variables . . . 15
2.2 ALE formulation of the governing equations . . . 16
2.2.1 Jacobian matrix and eigenstructure . . . 18
2.3 Node-pair finite volume discretization . . . 19
2.3.1 Metrics computation for a tetrahedral mesh . . . 21
2.3.2 Integrated numerical fluxes . . . 24
2.4 Time discretization . . . 26
2.4.1 Iterative implicit solver . . . 27
2.5 The IVC condition and the DGCL . . . 30
2.5.1 Interface velocity computation . . . 31

3 Conservative ALE scheme for adaptive meshes **35**
3.1 Continuum interpretation of 3D grid topology modifications . . . 35
3.1.1 Three steps procedure . . . 36
3.1.2 Edge creation . . . 42
3.1.3 Edge deletion . . . 42
3.1.4 Special care in node-pair modification . . . 43

3.2 ALE scheme with variable topology 44
3.2.1 ALE scheme for a new node 45
3.2.2 ALE scheme for a removed node 46
3.2.3 Conservative ALE governing equations for adaptive grids . 47

4 Three-dimensional mesh adaptation strategy 49
4.1 Metric construction 49
4.1.1 Error estimator computation 50
4.1.2 Target grid spacing 51
4.1.3 From the grid spacing to the metric map 52
4.1.4 Extension to different error estimates 54
4.2 Local mesh adaptation techniques 55
4.2.1 Node insertion 56
4.2.2 Node deletion 57
4.2.3 Edge swap 57
4.2.4 Node relocation 59
4.3 Mesh displacement 59
4.3.1 Robust strategy for large displacements 60
4.4 Summary of the computational procedure 61

5 Numerical simulations of reference ideal and non-ideal compressible fluid flows 67
5.1 Ideal-gas reference flows 67
5.1.1 Regular shock reflection in dilute gas conditions 68
5.1.2 Transonic flow around the oscillating NACA 0012 airfoil . . 69
5.2 Three-dimensional piston-induced shock-tube flows 70
5.3 Under-expanded nozzle jets of non-ideal compressible-fluids 75
5.3.1 Setup of numerical simulations 81
5.3.2 Assessment of mesh adaptation criteria 84

6 Three-dimensional flows around wings 89
6.1 Transonic steady flow around ONERA M6 wing 89
6.2 Infinite-span NACA 0012 wing 94
6.2.1 Steady simulation in the wing reference 94
6.2.2 Transonic flow around the oscillating NACA 0012 wing . . 95
6.2.3 Unsteady simulation in the laboratory reference 101
6.3 Finite-span NACA 0012 wing 101
6.3.1 Steady simulation 109
6.3.2 Unsteady simulation in the laboratory reference 110

7 Conclusions and future developments 117

Bibliography 128

List of Figures

2.1 Representative Van der Waals isentropes in NICFD regime for carbon dioxide . . . 16
2.2 Domain and boundary finite volumes definition for tetrahedral grid. 20
2.3 Domain and boundary metrics definition . . . 21
2.4 Orientation of the elemental contribution to the integrated normal 22
2.5 Extended node-pair structure for the node-pair i-k for the high-resolution integrated flux . . . 26
2.6 Volume swept by the elemental interface because of nodes displacement . . . 33

3.1 Three-steps procedure applied to the edge split in 2D . . . 38
3.2 Local character of the edge slit . . . 38
3.3 Three-steps procedure applied to the edge split . . . 40
3.4 Three-steps procedure applied to the edge collapse . . . 41
3.5 Sequence of modifications involving the same node-pair during the time step Δt^n . . . 44

4.1 Multi-passage strategy . . . 53
4.2 Edge split . . . 56
4.3 Element split . . . 57
4.4 Element collapse . . . 58
4.5 Edge swap . . . 58
4.6 Robust strategy for large displacements . . . 62
4.7 Computational procedure for unsteady problems . . . 65

5.1 Regular shock reflection results at different adaptation levels . . . 68
5.2 Pressure profile for the regular shock reflection . . . 68
5.3 Grid at different time steps for the transonic pitching airfoil . . . 71
5.4 Mach contour lines at different time steps for the transonic pitching airfoil . . . 72
5.5 Lift coefficient hysteresis of the transonic pitching airfoil . . . 73
5.6 Initial mesh and initial uniform solution for piston-induced shock-tube test case . . . 74
5.7 Centerline pressure and Mach profiles obtained with different mesh adaptation parameters for the piston-induced shock-tube problem . 76
5.8 Mesh at $t = 0.211$ obtained with different mesh adaptation parameters for the piston-induced shock-tube problem . . . 77

5.9 Mesh of piston-induced shock-tube problem at different times . . . 78
5.10 Pressure contour plot for piston-induced shock-tube problem at different times . . . 79
5.11 Solution of piston-induced shock-tube flow at center-line . . . 80
5.12 Weakly and highly under-expanded jet structure . . . 80
5.13 Operating conditions of under-expanded nitrogen jets in the thermodynamic plane . . . 83
5.14 Reference solution for the under-expanded nozzle jet test . . . 83
5.15 Comparison of different estimators for the under-expanded nozzle jet test . . . 84
5.16 Comparison of different compound estimators for the under-expanded nozzle jet test . . . 85
5.17 Solutions of the under-expanded nozzle jet after different adaptation steps . . . 86
5.18 Grid and Mach contour plot of the under-expanded nozzle jet using the estimator $\mathcal{H}(P) + \mathcal{H}(\rho)$ . . . 87
5.19 Increase of grid nodes and elements with adaptation steps . . . 87
5.20 Comparison of the solution of the under-expanded nozzle jet with ideal gas and van der Waals gas model . . . 88

6.1 Grid and Mach contour plots of the transonic ONERA M6 flow after different adaptation cycles . . . 91
6.2 Pressure coefficients for the ONERA M6 wing at $\mathrm{M}_\infty = 0.8395$ and $\alpha = 3.06°$ . . . 92
6.3 Fixed fine grid and solution for the ONERA M6 wing problem . . . 93
6.4 Pressure coefficients for the ONERA M6 wing at $z/b = 0.65$ . . . 93
6.5 Geometry of the computational domain for the test if infinite-span NACA 0012 wing . . . 94
6.6 Grid and Mach contour plots of the transonic steady flow around the infinite-span NACA 0012 wing after different adaptation cycles 96
6.7 Details of the grid for the transonic steady flow around the infinite-span NACA 0012 wing . . . 97
6.8 Detail of the computational grids for the pitching infinite-span NACA 0012 wing at different time steps . . . 98
6.9 Details of grid near the trailing and leading edge for the pitching infinite-span NACA 0012 wing . . . 99
6.10 Mach contour lines of the flow around the pitching infinite-span NACA 0012 wing at different time steps . . . 100
6.11 Mach contour plot of the flow around the pitching infinite-span NACA 0012 wing at different time steps . . . 102
6.12 Pressure iso-lines at three sections of the pitching infinite-span NACA 0012 wing at different time steps . . . 103
6.13 Pressure coefficient at three sections of the pitching infinite-span NACA 0012 wing at different time steps . . . 104
6.14 Lift coefficient hysteresis of the transonic pitching infinite-span NACA 0012 wing . . . 105
6.15 Integral value of the density over the domain for the transonic pitching infinite-span NACA 0012 wing . . . 105

6.16 Mesh during the unsteady simulation of the infinite-span NACA 0012 wing in the laboratory frame. 106
6.17 Pressure contour lines during the unsteady simulation of the infinite-span NACA 0012 wing in the laboratory frame. 107
6.18 Lift coefficient for the unsteady simulation of the infinite-span NACA 0012 wing in the laboratory frame 108
6.19 Integral value of the density over the domain for the unsteady simulation of the infinite-span NACA 0012 wing in the laboratory frame 108
6.20 Computational domain for the simulations of the flow around the finite-span NACA 0012 wing 109
6.21 Grid and Mach contour plots of the transonic steady flow around the finite-span NACA 0012 wing after different adaptation cycles . 111
6.22 Mesh during the unsteady simulation of the finite-span NACA 0012 wing in the laboratory frame. 112
6.23 Pressure contour lines during the unsteady simulation of the finite-span NACA 0012 wing in the laboratory frame. 113
6.24 Lift coefficient for the unsteady simulation of the finite-span NACA 0012 wing in the laboratory frame 114
6.25 Lift coefficients for a finite-span NACA 0012 wing in the laboratory frame using different error estimators 114
6.26 Iso-surface of vorticity magnitude for the simulation of the finite-span NACA 0012 wing in the laboratory reference frame 115

Chapter 1

Introduction

1.1 Motivation and goal

Computation Fluid Dynamics (CFD) proved to be an effective analysis and design tool in many research and industrial areas, especially in the aerospace one. Fundamental physics, numerical analysis and computer science are exploited to model the flow field at the required level of accuracy and to solve it with the available computational resources. Despite the improvements in numerical techniques and the steady rise of available computer power in recent decades, geometrically complex and three-dimensional moving-body problems still represent a challenge for the CFD community. Unfortunately, a large set of engineering problems of interest in aerospace falls exactly in this class. Moreover, it is often relevant for CFD analyses to capture peculiar flow features such as for example vortical structures or shock waves, whose exact location is not known a priori.

An example is represented by the flow field generated around the helicopter in forward flight, in which aerodynamics, flight dynamics, aeroelasticity and acoustics strongly interact. The superimposition of the advancing and rotational velocity of the blade determines the simultaneous occurrence of diverse physical phenomena and spatial characterized by different temporal scales, including transonic flow, shocks, reverse flow, dynamic stall and wakes. In addition to multidisciplinary aspect, rotor-craft simulations are extremely demanding from the numerical point of view. Since fluid-structure interaction is found to be one of the major source of noise and vibrations of helicopters, the accurate representation of the local flow features is required, along with the capability to deal with the arbitrary large deformation experienced by the solid boundaries.

Similar complexities stand also in numerical simulations of wind turbines. In order to make wind energy more competitive from an economical point of view, the actual trend is to increase the size of wind turbine and typical rotor diameter can be on the order of some tens of meters. On blades of this length, the aeroelastic effects have to be taken into account and bi-dimensional aerodynamic models, based on thin airfoil theory, cannot be used. Hence, the capability to deal with three-dimensional moving body is mandatory for CFD codes also in these applications.

A further example of challenging simulation in the aeronautical field concerns the deployment of control surfaces. In the transonic regime, the modification of aerodynamic characteristics may originate highly non-linear effects such as shock waves and separated flows. Therefore, the accurate prediction of the flow field around deployed control surfaces requires, in addition to the necessity of complying with body deformation, also the capability of locally increasing grid resolution to capture transient flow structures.

Additional examples of time dependent geometries include piston motion and valve operation in internal combustion engines, rotor-stator interactions in turbomachinery and store separation from aircrafts, to name a few. Although characterized by problem-dependent requirements, the CFD simulation of all the cited flow fields calls for a robust and accurate strategy to handle complex moving geometrical configurations and, at the same time, to locally modifying the grid resolution according to the solution itself.

To this end, the present work aims at developing an innovative adaptive scheme, well-suited to unsteady, three-dimensional flow simulations around complex time-dependent geometries, with the capability of capturing the details of the flow features including shock waves and shear layers. Thanks to an innovative interpretation of grid connectivity changes, the solution on the new grid is recovered by exploiting the arbitrary Lagrangian-Eulerian (ALE) formulation of the Euler equation without any explicit interpolation.

In the next two sections, the background of the exploited techniques is given. In particular, Section 1.2 provides an overview of the ALE formulation and of some boundary movement methods. Section 1.3 presents a brief summary about mesh adaptation techniques. The objectives and the structure of the thesis are then outlined in Section 1.4.

1.2 Numerical solution of the flow equations on moving boundaries

Most current CFD techniques require that the flow governing equations are spatially discretized over a computational grid (or mesh), whose quality and resolution strongly influence the solution accuracy. When solving the governing equations on a moving grid, as mandatory in moving boundary problems, two fundamental aspects should be considered. First, the formulation of the governing equations has to be modified in some way to account for the relative motion of the grid with respect to the fluid. Second, if a motion is prescribed to the boundaries of the grid, the interior points should be relocated to maintain the required mesh quality.

This section gives an overview of the Arbitrary Lagrangian-Eulerian (ALE) method, which is a popular framework used to tackle moving boundary problems, and of some among the several strategies proposed to drive grid modification, with particular reference to those capable of dealing with large displacement.

1.2.1 Arbitrary Lagrangian-Eulerian framework

The numerical simulation of fluid dynamics and continuum mechanics problems often requires to deal with strong internal distortions of the fluid domain and,

at the same time, to precisely follow the boundaries and the interfaces of multi-material systems. The capability to cope with both requirements is determined by the choice of the kinematic description, which expresses the relationship between the deforming continuum and the computational mesh on which the governing equations are solved. The most natural descriptions are the Lagrangian and Eulerian viewpoints.

In the Lagrangian description, the initial unmodified configuration is chosen as the reference one and, during the continuum deformation, each grid point follows the material point to which it is associated initially. This constraint allows to easily track boundaries and interfaces, because they are always associated to the same points. However, Lagrangian algorithms are not suited to follow large deformations, because of excessively stretched or tangled elements which reduce accuracy and may invalidate the mesh.

In the Eulerian description, possibly the most popular in fluid dynamics, the grid nodes are held fixed and the reference configuration is the current one. In this approach, each grid cell is crossed by different material points as the continuum deforms. Eulerian algorithms can easily deal with large distortions of the fluid but the tracking of moving boundaries and interfaces is difficult. Moreover, the relative motion between the continuum and the grid results in advective (or convective) terms that have to be taken into account in the governing equations.

The Arbitrary Lagrangian-Eulerian (ALE) formulation can be regarded as a generalized kinematic description that combines the Lagrangian and the Eulerian approach. The computational mesh is neither fixed in space nor attached to the material particle but it moves independently, exploiting the advantages and reducing the drawbacks of both approaches. As in the Eulerian description, additional advective terms have to be included in the governing equations.

A classification of ALE algorithms can be done according to the handling of the convective terms. A possibility is represented by the so-called decoupled ALE approach, in which a Lagrangian step is followed by a convective step. The governing equations are first enforced in a pure Lagrangian framework and no mass flux is exchanged between grid cells. Then, the mesh is modified and the solution is remapped over the new grid. In this latter phase, convective fluxes must be computed to account for the apparent displacement of fluid between cells as the mesh moves and the solution must be conservatively transferred over the grid. Moreover, the remapping step has to preserve the monotonicity of the solution. These requirements can be enforce during the remap step properly, driving, for instance, the remap according to an estimation of the mass exchanged between cells [95], or performing a repair step after an interpolation [79].

The second category consists of the ALE algorithms that solve the fully coupled equations involving both material and mesh velocities. No distinction between the mesh update and remap phase is made, since they are performed simultaneously. These algorithms can be viewed as an extension to moving meshes of the classical Eulerian formulation, which already includes conservative terms. The main advantages of this approach is that no explicit interpolation is required, since the remap is embedded in the governing equations. Therefore, conservativeness and monotonicity are inherited by the original scheme. Many different techniques have been proposed in the open literature and in recent years the coupled approach has been extended also to high-order schemes [91, 45, 54, 98].

Besides the need of complying with the movement of the domain boundary,

the capability of the ALE scheme to deal with moving grid points can be exploited to perform grid regularization and mesh adaptation. The former technique aims at keeping the computational grid as regular as possible and grid modifications are driven basically on geometrical considerations. The boundaries, whose motion is possibly known at prior or, at least partially, unknown, are described by a Lagrangian formulation, while the displacement of internal points can be determined by several interpolation techniques, such as, for instance, mesh smoothing, transfinite mapping, optimization of local mesh quality [132, 65, 58] or the dynamic grid techniques described in the next subsection. On the contrary, mesh adaption strategy aims at optimizing the grid according to the solution. On the base of an error indicator, the grid points are gathered in the regions of the grid with strong solution gradients. However, note that the total number of points and the grid connectivity cannot be modified in the standard ALE formulations.

1.2.2 Small boundary displacement: dynamic grids

Unsteady numerical simulations often require to update the computational domain to follow the body motion or the deformation of some of its surfaces. The so-called dynamic grid methods can be used to efficiency deform the grid keeping fixed the connectivity and changing only the positions of the grid nodes. The underlying idea is to distribute the deformation occurring at the boundary among internal grid elements depending on their shape and volume. Smallest elements, which are usually gathered near the moving body, are subject to a minor distortion, while the largest ones accommodate the most part of the deformation. This goal can be achieve by moving each point individually in the so-called point-by-point schemes [31], or by exploiting the mesh connectivity, as for instance in the elastic analogy or by solving a set of partial differential equations (PDE).

In the elastic analogy the spatial domain is treated as an elastic material whose boundaries are subject to an imposed deformation which is propagated to the internal part through elastic forces [13]. The elastic properties of the grid cells are defined as a function of their shape and volume. A similar approach is the spring analogy, proposed by Batina [12], which considers each grid node connected to its neighbors by a fictitious spring whose stiffness is inversely proportional to the their distance. The displacement field is obtained by imposing the static equilibrium at each node. This technique was improved by Farhat and collaborators [43] who included also a torsional spring to avoid element tangling.

In the PDE technique the displacement of the internal grid nodes is computed by solving an elliptic problem. For instance, a Lagrangian smoothing with variable diffusivity based on the distance from moving bodies or on the element volume can be used to avoid large distortions, and therefore an accuracy loss, in the most critical regions [93, 27]. The use of equations of order higher than the second for the internal deformation permits a finer control on grid quality, but at a higher computational cost [69].

The main drawback of all these methods is the computational cost, which can become extremely high for large deformations or for unsteady problems that require several mesh updates per time-steps. Moreover, the fixed-topology constraint limits the displacement that the mesh is capable to handle without invalidating the mesh elements or jeopardizing the accuracy of the solution.

1.2.3 GCL: an essential constraint for deforming meshing

When solving the flow problems over deforming grids, a Geometric Conservation Law (GCL) needs to be satisfied to avoid errors due to mesh motion. Proposed first by Thomas and Lombard [119], the GCL states that a numerical scheme designed to enforce the governing equations over moving grids must compute exactly a uniform steady-state flow without introducing any perturbations. This requirement is usually fulfilled by an appropriate computation of the geometric quantities that involve grid positions and velocities, like for instance the cell volume or the integrated convective fluxes.

This issue has been addressed since the first dynamic grid methods [12]. The success of the ALE formulation led to a thorough investigation of the GCL condition and apparently contrasting results have been reported. The discrete counterpart of the GCL, the so called Discrete Geometric Conservation law (DGCL), was introduced by Loseille and Farhat [86], who proposed a general space-time discretization method to satisfy the DGCL for finite volume and finite element schemes. Although Guillard and Farhat proved that, for a p-order time-accurate scheme, meeting the DGCL condition is a sufficient condition to be at least first-order time-accurate on a moving mesh [63], Geuzaine and co-authors demonstrated that the DGCL is neither a necessary nor a sufficient condition for an ALE numerical scheme to preserve on moving grids its order of time-accuracy established on fixed grids [56].

On the other hand, respecting the DCGL is a necessary and sufficient condition for preserving on moving mesh the non-linear stability of the underlying time-integration scheme, namely the absence of spurious oscillations and conservation of monotonicity [44]. Indeed, several studies have shown that non-DGCL-compliant schemes lead to spurious oscillations and require smaller time steps [48, 98].

Therefore, it is generally accepted that the fulfillment of the DCGL helps in improving the accuracy of the numerical scheme, in preserving non-linear stability and in preventing spurious oscillations. This is of paramount importance in fluid-structure interaction problems, where a violation in the conservation of the flow quantities (especially momentum and energy) may lead to erroneous calculation of fundamental quantities [42], such as the flutter velocity. An updated and comprehensive review of the literature on this subject can be found in [41].

1.2.4 Large boundary displacement

In many of the problems discussed in Section 1.1, the boundaries of the computational domain experience arbitrarily large displacement so that their accurate tracking becomes a fundamental requirement for the CFD code. However, as observed above, fixed-connectivity grids fail in dealing with large deformations. Therefore a different strategy has to be used.

A possible solution consists in moving the mesh keeping the topology fixed until the quality falls below a certain threshold. The computational domain is then completely re-meshed and the solution is interpolated on the new grid. However, the additional interpolation error introduced by a frequent regeneration of the grid and the quite considerable computational cost make this technique seldom convenient. Furthermore, the interpolation of the solution between two computational grids can introduce numerical oscillations and undermine the conservative

properties and the accuracy of the numerical scheme.

More efficient methodologies involve a completely different approach. Instead of meshing the computational domain as a whole, the latter is partitioned into a set of simpler blocks that can move with respect to one another. The idea is exploited in the Chimera method, originally developed by Benek *et al.* to simulate the compressible transonic flow field around the complete Space Shuttle [14]. This method is well-suited to perform numerical simulation of rotor-crafts and turbo-machinery, because each grid can rigidly rotate and translate independently. Moreover, multiple sub-domains can be used to prescribe different spatial resolutions according to the scales of the phenomena of interest and the domain decomposition can be exploited to perform parallel simulations.

The main difficulty concerns the communication between different sub-domains. Indeed, the governing equations are solved separately in each sub-domain and a background overlap grid is used to match the solutions. A non-trivial time-consuming tagging procedure is required to locate the interface points on the background grid, which should also guarantee a suitable overlap for all possible displacement. Moreover, the simple interpolation of the variables between different grids does not guarantee that the solution scheme is conservative, i.e. that mass, momentum and energy are conserved across the interface. In compressible flows, conservativeness is a crucial requirement to correctly propagate discontinuities and unphysical solutions can be obtained if it is not met. Although many efforts have been made to develop conservative schemes, the fulfillment of the global conservation for Chimera overlapping methods still remains an open problem, especially in three spatial dimensions [117, 126].

A similar technique is represented by the so-called sliding meshes, in which neither overlaps or gaps between different sub-domains are permitted. A sliding surface is defined as the interface between non-conformed sub-grids. If this interface is planar, a conservative interpolation can be exploited by splitting it into conformed sub-faces [110, 47, 50]. However, the requirement to exactly match sub-grids along the boundaries is not easy or possible to meet for complex domains and/or relative body motions, like for instance load separation or deployment of control surfaces [117].

A further class of methods well-suited to deal with large boundary displacements are the immersed/embedded boundary methods [101]. The mesh generation issue is simplified because the computational domain is discretized by a non-body-fitted mesh. The governing equations are then enforced in an Eulerian formulation without moving the grid. Although simple and consistent, these methods are not very accurate near the boundaries and the enforcement of boundary conditions is complicate [1]. Therefore, immersed boundary methods cannot be easily applied if high accuracy is required in the close proximity of complex geometries.

Furthermore, the use of adaptive unstructured space-time meshes has been recently proposed by Wang and Persson [127] to handle large grid deformations in two dimensions. However, to author's knowledge, it has not been extended to three-dimensional problems.

A different approach is applied in this thesis. Similar to the dynamic grid strategy, the mesh is deformed as much as possible with fixed topology. Then, local grid modifications are performed, including connectivity changes. Following the methods proposed for bi-dimensional problems by Guardone *et al.* [59], the

local modifications are represented as continuous grid alterations within the ALE framework and no explicit interpolation is required. This allows to preserve the scheme properties, especially conservativeness and monotonicity. Moreover, in contrast to the other above mentioned methods, the implementation of multi-step time schemes does not require to store all the previous grids but instead only local modifications history have to be saved.

1.3 Mesh adaptation techniques

Mesh adaptation is widely used in CFD simulations to tackle diverse flow features characterized by different spatial scales and in all problems where the general location of relevant flow characteristics is not known a priori [10, 92]. The grid spacing has to be related to the behavior of the solution in an efficient way to avoid inaccurate predictions of integral quantities (like aerodynamic coefficients) or of local flow features (shock waves, contact discontinuities, etc.). Solution-dependent adaptive grid techniques can be iteratively employed to achieve this goal without user intervention. To enhance the accuracy of the overall solution without excessively increase the computational burden, grid resolution can be increase where the discretization error is estimated to be large. Hence, the key of the success or failure of mesh adaptation techniques is strictly related to the criteria used to identify precisely where grid modifications are required.

Furthermore, when dealing with large boundary movements, if simple deformation is not sufficient to perform the whole displacement, local mesh adaptation techniques can be used to monitor mesh quality and to avoid invalid elements. In these situations, elements to be modified are determined on the basis of deformation measures, shape or aspect ratio [9].

A possible classification of the different mesh adaptation strategies concerns the mechanism used to alter the mesh. Indeed, three main classes are usually defined: the p-refinement methods, the r-refinement methods, the h-refinement methods. The first method, mainly used in finite-element schemes rather than in finite-volume schemes, does not alter the mesh and it performs refinement by increasing the order of accuracy of the polynomial in each element and it is often used in combination with the latter method, in the so-called hp-refinement [8, 55, 38]. On the contrary, the r and the h-refinement methods alter the grid keeping the connectivity fixed or modifying it, respectively, as described in more detail in later sections. In the present work, p-refinement methods are not used and therefore they are omitted by any further description.

A crucial step in mesh adaptation is the re-construction of the solution on the adapted grid. The solution computed on the old mesh has to be transfered to the new one, but grid connectivity has changed. Classical techniques perform an interpolation of the old solution on the new grid. As already pointed out, this operation can undermine the accuracy of the solution and the conservation properties of the scheme.

In this section, the two fundamental aspects central to any mesh adaptation strategy are investigated. Indeed, in Subsection 1.3.1 a review of some adaptation criteria proposed in the literature is given, then Subsection 1.3.2 briefly describes the possible mesh modification strategies. Finally, the peculiarities of mesh adaptation in unsteady simulations are outlined.

1.3.1 Adaptation criteria

A key point in mesh adaptation algorithms is the definition of a suitable criterion, either integral or local, to drive grid alteration. Since an exact characterization of the error is usually impossible, several error estimators and criteria for feature detection have been proposed.

The simplest indicators are based on gradients or undivided differences of some relevant flow variable, such as the density, the pressure, the Mach number, or a combination of them. These methods implicitly assume that the largest errors occur in high-gradient regions and have proved to be successful in various problems [28, 74, 129, 23].

More complex indicators can be built through an estimate of the error introduced by the underlying discrete numerical approximation. For instance, Richardson's extrapolation technique has been used to estimate the truncation error, namely how well the discrete model approximates the governing equations throughout the domain [15, 16, 3]. Also interpolation-based error estimators error have been used to generate scalar and metric indicators to describe the optimal grid spacing [108, 130, 22, 64]. Error estimator can be also obtained by evaluating the local residual [118].

A possible drawback of local error estimators is that they do not take into account the propagation of errors through the domain and their impact on the output quantities. Moreover, continuous local refinement of the strongest features of the flow does not necessarily guarantee a corresponding reduction in the overall discretization error and, sometimes, this may lead to incorrect results [129, 124]. To overcome the local character of previous indicators, integral output-based estimators based on the solution of the adjoint flow problem have been developed [124]. The underlying idea of this technique is that, at the cost of computing the solution of the adjoint problem and the storage of the Jacobian (and its computation if not required by the numerical scheme), it is possible to control the accuracy of the computed output of interest using mesh adaptation. A detailed review of this class of estimators can be found in [46].

When features of interest such as shocks, boundary layers and discontinuities exhibit large variations along one direction and less significant changes along the other ones, anisotropic adaption is recommended to increase grid resolution only along the relevant direction, limiting the number of new grid points. Since to deal with anisotropic grids not only the size of the elements is required but also their shape and orientation, a bi-dimensional or three-dimensional error estimator has to be built in the form of a metric tensor [53, 30, 37]. This metric map can be based on the Hessian of a relevant flow variable, which provides an estimate of the interpolation error, and determines a Riemann space in which the length of the grid edges is measured. The goal of anisotropic adaptation is to obtain a unit mesh, i.e. a mesh in which all edges measured in the Riemann space defined by the metric map have a unit length [22].

In this work, local error indicators based on the derivatives of flow variables are chosen because of their easiness of computation. To limit the possibility of refining only around the strongest features, a multiple evaluation of the error estimation is used, as suggested by Aftosmis [2], to detect also weaker variations in the flow variables of interest.

1.3.2 Adaptation strategy

A possible strategy to meet the computed optimal spacing consists in regenerating completely the grid from scratch [107]. To accomplished this goal, many automatic grid generation techniques are available and this subject has been thoroughly covered [120, 52, 97]. However, in many case of practical interest, the modifications are required in small regions of the grid and a complete re-meshing can be very inefficient, especially in unsteady simulations. Even if re-meshing can be applied locally, i.e. the regions that require to be altered are identified, deleted and re-meshed using standard grid generation methods such as for example the advancing-front method or Delaunay triangulations, its applicability and efficiency are strictly limited by the shape and dimension of the regions to be re-meshed.

A more efficient alternative consists in improving the grid spacing by an appropriate sequence of local geometrical and topological modifications, such as node insertion and deletion, edge or face swap and point relocation. Several adaptive algorithms that exploits this possibility have been proposed, like, for instance [130, 37, 9, 89, 33, 29].

Classical grid enrichment techniques consist in element subdivision and were first implemented by Berger and Jameson [15] and Dannenhoffer and Baron [28]. Easy to implement and computationally efficient, these techniques are suitable for triangular, tetrahedral and mixed-element grids, provided that a set of possible refinement patterns is defined [92, 130]. A drawback of this technique is the possibility to generate hanging nodes, i.e. non-conforming connectivities between adjacent elements, which are usually not permitted by flow solvers. Hence, compatibility checks have to be performed or additional elements have to selected for refinement to restore the correct local topology [73, 99]. To avoid an excessive increase in the number of grid points, the node in low error regions can be deleted by edge collapse or by using in the inverse way the pattern defined for refinement.

When mode displacement, element subdivision and edge collapse lead to ill-shape elements, their quality can be recovered by edge or face swap. This techniques aims at modifying the connectivity to improve mesh topology, keeping unaltered the position and number of grid points. Proposed by Lawson [84] as an algorithm to for transforming triangulation, edge swap can be exploited to dramatically improve the overall mesh quality.

A different strategy for adding node relies on the Delaunay triangulation. To insert a new point in a certain location, the elements having a circumsphere that includes the new point are deleted to create a suitable cavity around the new node. The cavity is then triangulated using a simple procedure. Although in two dimensional grids this procedure usually results in grid elements of acceptable quality, in 3D it may lead to badly-shaped elements, like slivers. To prevent this, a minimal volume requirement can be imposed [33].

Finally, the global technique called r-refinement or vertex relocation can be used with two aims: to gather points toward regions where a finer grid spacing is required and to increase the quality of the grid elements, according to a prescribed quality function, usually dependent on the element geometry. The main advantage is that grid topology is held constant and therefore no complex data structures to handle connectivity changes are required. Unfortunately, as already pointed out in the previous section, large boundary displacements usually lead to

very poor element quality and additional precautions has to be taken to prevent element entanglement or inversion.

1.3.3 Unsteady mesh adaptation

In steady-state adaptive-grid simulations, an iterative procedure is usually performed, namely the information obtained from the solution on a coarser grid is used to drive the successive grid enrichment. Few adaptation phases are usually required to converge toward a fixed point for the pair formed by the mesh and the solution.

When unsteady problems are tackled, a complete mesh adaptation phase, as in the steady case, is usually performed every several time steps, while only a r-refinement is eventually performed every time step to cope with the boundary displacement, see for instance [111, 105, 90]. Differently from the steady case, the efficiency is much more important than optimality, thus mesh coarsening is essential to keep the number of grid points under control. Unfortunately, these classical strategies are characterized by a time shift between mesh and solution, because mesh adaptation is driven by the estimator built on the solution computed at the previous time step. Therefore no prediction in the evolution of the time-dependent flow features is usually carried out.

A different strategy has been developed by Alauzet *et al.* [6, 5], who propose to build a transient fixed-point problem by introducing in the main loop, an additional adaptation loop at each time step. Starting from the solution computed at time t, an iterative mesh adaptation procedure is performed to converge towards a fixed point for the time $t + \Delta t$. Furthermore, to improve the capability of capturing transient solutions, they developed also a time-dependent metric defined by means of an intersection procedure in time.

1.4 Thesis goals and outline

This thesis proposes a novel approach to solve the finite volume formulation of the unsteady inviscid compressible governing equations over three-dimensional dynamic grids. Mesh adaptation is exploited to successfully deal with arbitrarily large movements of solid boundaries and to capture phenomena with different spacial scales. The innovative interpretation of the local connectivity changes as continuous deformations of the finite volumes that compose the computational domain, proposed by Guardone and collaborators [59, 71], is here extended to the three-dimensional case.

Modifications due to grid adaptation are then interpreted as continuous element deformations within the Arbitrary Lagrangian-Eulerian framework and hence any explicit interpolation between different grids is avoided. Therefore, the conservation and accuracy properties of the ALE scheme are preserved. The lack of the interpolation between grids at different time steps allows to easily implement high-order multi-step time schemes, like for instance Backward Differentiation Formula. Moreover, the computation of the geometrical quantities related to mesh motion is carried out in a way that automatically satisfies the DGCL, even if topological modifications occur.

The two-dimensional version of the proposed conservative adaptive scheme is already available in the software FlowMesh, currently under development at the

Department of Aerospace Science and Technology of Politecnico di Milano [70]. Therefore, the three-dimensional strategy proposed in the present work is implemented in the same software. Moreover, to overcome the difficulties of performing local grid modifications in three dimensions, the external open-source library Mmg3d [36] is linked to the flow solver.

The validity of the proposed approach is proved by steady and unsteady simulations of reference 2D and 3D problems, both in the ideal gas region and within the so-called non-ideal compressible-fluid dynamics regime, namely in the proximity of the liquid-vapor critical point and saturation curve [25]. Then three-dimensional accurate simulations of aeronautical interest are carried out by exploiting the proposed conservative adaptive interpolation-free strategy.

The outline of the thesis is as follows.

Chapter 2: *Finite Volume ALE scheme for three-dimensional flows*
In this chapter, the space discretization and time integration of the governing equations are detailed. The ALE formulation of the compressible Euler equations is discretized by means of a finite volume scheme over an unstructured grid, or more precisely over the corresponding median-dual mesh. The computation of all involved geometrical quantities and the numerical fluxes is thoroughly explained. To enforce monotonicity near discontinuities in the solution, a flux limiter approach is used to combine a second-order flux approximation with a first-order one. With regard to the time integration, a standard Backward Differentiation Formulae (BDF) scheme is adopted, together with an implicit dual-time stepping scheme. Particular care is taken in the computation of the grid velocity that is embedded in the system of governing equations to automatically satisfy the DCGL in case of moving grids.

Chapter 3: *Conservative ALE scheme for adaptive meshes*
The finite volume ALE scheme described in the previous chapter is here extended to adaptive grids with local connectivity changes. Thanks to a sequence of fictitious collapses and expansions of adjacent elements, grid modifications are treated as continuous deformations of the finite volumes that compose the computational domain. The complete system of governing equations for adaptive grids is finally derived.

Chapter 4: *Three-dimensional mesh adaptation strategy*
The complete mesh adaptation strategy used in this work is detailed in this chapter. Besides the standard mesh deformation technique based on elastic analogy, all the considered grid modifications are detailed. Thanks to the library Mmg3d, edge split, edge collapse, element split and node insertion by Delaunay triangulation are mixed together to obtained a mesh conforming to the given metric. The target grid resolution is computed according to an error estimator computed from the solution field obtained in a previous computation, complemented with a number of grid quality constraints. Finally, the complete adaptation strategy is summarized, focusing on the linking between the flow solver and the mesh adaptation library.

Chapter 5: *Numerical simulations of reference ideal and non-ideal compressible-fluid flows*
This chapter aims at assessing the reliability of the proposed computational strategy. First, some standard two-dimensional problems of ideal gas flows are tackled, including the shock reflection problem and an oscillating airfoil at transonic speed, to establish the capability of capturing the relevant flow features. Then, three-dimensional simulations of piston-induced shock-tube flows are performed and the numerical results are compared to the analytical solution to verify their accuracy. Finally, mesh adaptation strategy and criteria are tested also in the non-ideal compressible-fluid dynamic regime by computing a nozzle flow with non-monotone Mach variation in the divergent part and an under-expanded jet of nitrogen at nozzle discharge.

Chapter 6: *Three-dimensional flows around wings*
In this chapter, the conservative adaptive ALE scheme is applied to problems of aeronautical interest. A steady simulation of the transonic flow around the Onera M6 wing is performed and mesh adaptation is exploited to accurately capture the peculiar lambda shock. Then, several simulations of the flow field around an infinite-span NACA 0012 wing are carried out. The capability of the proposed strategy of dealing with large boundary movements is assessed by means of a steady simulation in the wing reference frame and an unsteady simulation in the laboratory reference frame. The transonic flow around the oscillating wing is then computed and compared with the bi-dimensional results. Finally, the steady and unsteady flows around a finite-span NACA 0012 wing are also tackled.

Chapter 2

Finite Volume ALE scheme for three-dimensional flows

This Chapter opens with the integral formulation of the Euler equations in both the Eulerian and Arbitrary Lagrangian Eulerian framework, completed by an appropriate thermodynamic model of the fluid. Section 2.3 describes thoroughly the finite volume discretization of the Euler equations over an unstructured grid, and it reports also on the computation of the required geometrical quantities related to the computational grid. Then, Section 2.4 explains the discretization and integration in time of the governing equations. Finally, the computation of the interface velocity in a way that guarantees that the GCL is automatically fulfilled is presented in Section 2.5.

2.1 Euler equations

The Euler equations describe compressible inviscid hydrodynamics and in many cases of practical interest they provide an accurate representation of the flow field, except in a small, boundary layer region close to the solid bodies. The approximations introduced when deriving this model are valid under the continuum hypothesis, namely for sufficiently large fluid density, and provided that thermodynamic as well as chemical equilibrium can be assumed. Moreover, neglecting the effects of viscosity and thermal conductivity is acceptable for flows around streamlined bodies and characterized by a high Reynolds number, namely for flows with thin boundary layers and no separation regions, and with a moderate Mach number, i.e. when hypersonic effects are irrelevant. Therefore, Euler equations can be used to model compressible-fluid flows, shock reflections and collisions, blast waves, flows past aerodynamic bodies and related physical phenomena. In aerodynamics, the Euler equation can predict only some features, like the wake and the induced or wave components of the drag, but not the overall efficiency of an aircraft. Since in this work the interest is addressed to the

predictions of pressure distribution on the surface of aerodynamic bodies, the Euler equations are chosen as the appropriate mathematical model.

The integral form of the unsteady Euler equations for a fixed control volume $\mathcal{C}$ contained in the domain Ω reads

$$\frac{d}{dt}\int_{\mathcal{C}} \mathsf{u}\, d\boldsymbol{x} + \oint_{\partial\mathcal{C}} \mathbf{f}(\mathsf{u}) \cdot \boldsymbol{n}\, d\boldsymbol{s} = 0 \tag{2.1}$$

where $\boldsymbol{x} \in \Omega \subseteq \mathbb{R}^3$ is the position vector, $t \in \mathbb{R}^+$ is the time, u is the vector of conservative variables, $\mathbf{f}(\mathsf{u})$ is the inviscid flux and $\boldsymbol{n}(\boldsymbol{s}) \in \mathbb{R}^3$ is the outward unit vector normal to the boundary $\partial\mathcal{C}$ of the control volume $\mathcal{C}$, which is function of the boundary coordinate $\boldsymbol{s} \in \partial\Omega \subseteq \mathbb{R}^2$.

The vector of conservative variables $\mathsf{u} : \Omega \times \mathbb{R}^+ \to \mathbb{R}^5$ is defined as

$$\mathsf{u} = \left[\rho,\, \boldsymbol{m},\, E^{\mathrm{t}}\right]^{\mathrm{T}},$$

where ρ is the density, $\boldsymbol{m}$ the momentum and E^{t} the total energy. The flux function $\mathbf{f} : \mathbb{R}^5 \to \mathbb{R}^5$ is defined as

$$\mathbf{f}(\mathsf{u}) = \begin{bmatrix} \boldsymbol{m} \\ \boldsymbol{m} \otimes \boldsymbol{m}/\rho + \Pi(\mathsf{u})\mathbb{I}^3 \\ \left[E^{\mathrm{t}} + \Pi(\mathsf{u})\right]\boldsymbol{m}/\rho \end{bmatrix} \tag{2.2}$$

where $\Pi(\mathsf{u}) = \Pi(\rho, \boldsymbol{m}, E^{\mathrm{t}})$ is the pressure function, which depends on the thermodynamic model, and $\mathbb{I}^3$ is the 3×3 identity matrix. Equation (2.1) states that the rate of change of the total amount of the conserved vector u contained in the control volume $\mathcal{C}$ must balance the net flux of $\mathbf{f}$ through the boundary $\partial\mathcal{C}$. To make the governing equations complete, suitable initial and boundary conditions must be specified [87].

2.1.1 Thermodynamic model

A thermodynamic model is required to express the pressure in terms of conservative variables, which can be easily provided as a set of two compatible equations of states (EoS), usually a thermal and a caloric one. The perfect gas equation is the simplest analytical EoS and it is based on the hypothesis that the gas is composed by identical molecules of negligible dimensions (considered as points) which do not exert interaction forces between each other. If also the assumption of constant specific heat is made, the polytropic ideal gas model is obtained, which provides accurate predictions in numerous aerodynamic applications.

According to the ideal gas model, the isotherms in the volume-pressure (v-P) thermodynamic plane are hyperbolas, i.e., $P(T, v) = RP/v$, where v is the volume for unit mass, T is the temperature and R is the gas constant. The fluid state fulfills the assumption of gas ideality only if the specific volume is large enough, namely within the so-called dilute region. Approaching the vapor-liquid saturation curve and, in particular, the critical point the thermodynamic behavior of the substance deviates significantly from the ideal one. In this regime, called non-ideal compressible-fluid dynamics regime, transport and physical properties of the fluid exhibit a peculiar dependence on the pressure and temperature, which therefore require the inclusion of more complex EoS. The polytropic van der Waals model is the simplest thermodynamic model capable of taking into account,

Polytropic ideal gas	Polytropic van der Waals gas
$P(T,v) = \frac{RT}{v}$	$P(T,v) = \frac{RT}{v-b} - \frac{a}{v^2}$
$e(T,v) = \frac{T}{\delta}$	$e(T,v) = \frac{T}{\delta} - \frac{a}{v}$
$\Pi(\mathsf{u}) = \delta \left[E^{\mathrm{t}} - \frac{\lvert\boldsymbol{m}\rvert^2}{2\rho} \right]$	$\Pi(\mathsf{u}) = \delta \frac{E^{\mathrm{t}} - \lvert\boldsymbol{m}\rvert^2/(2\rho) + a\rho^2}{1-b\rho} - a\rho^2$

Table 2.1: Polytropic ideal gas and van der Waals thermodynamic models. δ is the inverse dimensionless specific heat at constant volume, namely, $\delta = R/c_v$, which is constant under the polytropic assumption. The van der Waals constants a and b are specifics of the fluid and are related to the critical temperature T_{C} and pressure P_{C} as follows: $a = 27/64\, R^2\, T_{\mathrm{C}}^2/P_{\mathrm{C}}$ and $b = 1/8 R\, T_{\mathrm{C}}/P_{\mathrm{C}}$.

at least qualitatively, these non-ideal compressible-fluid effects [25]. As shown in Figure 2.1, the isotherms of van der Waals fluids tend to ideal-gas hyperbolas at high temperature, but become steeper in the non-ideal compressible-fluid region.

In the present work the polytropic ideal and van der Waals models are used to derive the functional form of the pressure function $\Pi(\mathsf{u})$. The thermal and caloric EoS in both models are reported in Table 2.1, along with the pressure functions. A complete thermodynamic model allows to compute all the thermodynamic properties of the gas. For instance, the speed of sound, which is defined as

$$c^2 = \left(\frac{\partial P}{\partial \rho}\right)_s = -v^2 \left(\frac{\partial P}{\partial v}\right)_s ,$$

can be computed as

$$c^2 = -v^2 \gamma \left(\frac{\partial P}{\partial v}\right)_T , \tag{2.3}$$

where $\gamma = c_p/c_v$ is the ratio between the specific heat at constant pressure and the one at constant volume.

2.1.2 Non-dimensional variables

Dimensionless variables are introduced to properly scale the solution variable and to reduce the round-off errors in the computations. Starting from a reference pressure and temperature, P_{ref} and T_{ref} respectively, such as for instance the critical values or the ones of the free-stream. A possible consistence set of reference values for the conservative variables can be obtained from dimensional analysis as follows

$$\rho_{\mathrm{ref}} = \frac{P_{\mathrm{ref}}}{RT_{\mathrm{ref}}} , \qquad m_{\mathrm{ref}} = \sqrt{\rho_{\mathrm{ref}} P_{\mathrm{ref}}} = \frac{P_{\mathrm{ref}}}{\sqrt{RT_{\mathrm{ref}}}} , \qquad E^{\mathrm{t}}_{\mathrm{ref}} = P_{\mathrm{ref}} .$$

The remaining dimensionless quantities can be derived using the above variables. For instance, defining also a reference length l_{ref}, a reference time can be defined as

$$t_{\mathrm{ref}} = \sqrt{\frac{\rho_{\mathrm{ref}}}{P_{\mathrm{ref}}}}\, l_{\mathrm{ref}} = \frac{l_{\mathrm{ref}}}{\sqrt{RT_{\mathrm{ref}}}} .$$

The non-dimensional counterparts of the governing equations (2.1) and the flux function $\mathbf{f}(\mathsf{u})$ have the same functional form and therefore they are simply

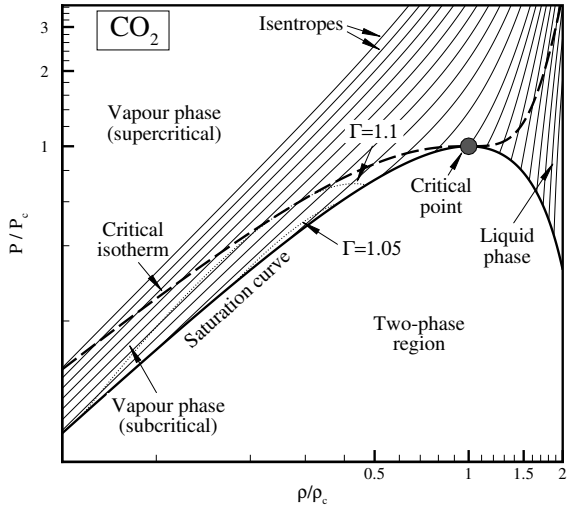

Figure 2.1: Representative isentropes and liquid-vapor saturation curve in the reduced ρ-P plane in the non-ideal compressible-fluid region for carbon dioxide under the polytropic van der Waals gas. Reduced variables are obtained by dividing dimensional ones by their corresponding critical point values. The critical isotherm, the critical point and iso-lines of the fundamental derivative of gasdynamics [121] are also shown.

evaluated using the non-dimensional variables, obtained e.g. as $\hat{P} = P/P_{\text{ref}}$. A slight modification occurs in the equation of state, because the gas constant R is chosen as reference variable both for the gas constant (itself) and the specific heats c_v and c_p. Indeed, the equation of state of the polytropic ideal gas simplifies to $\hat{P} = \hat{\rho}\hat{T}$. In the following, all the variables are assumed to be dimensionless. However, the "hat" symbol $\hat{}$ is not used in the following since it can be safely dropped without causing confusion.

2.2 ALE formulation of the governing equations

The Arbitrary Lagrangian-Eulerian formulation is now applied to solve the Euler equations over possibly moving control volumes. To this purpose, two different configurations are identified: the reference ALE domain $\Omega_{\boldsymbol{X}}$ and the current grid configuration $\Omega_{\boldsymbol{x}}(t)$, which corresponds to the Eulerian reference frame. Following the work of Donea *et al.* [39], the mapping between these configurations is defined as

$$\begin{aligned} \boldsymbol{\varphi} : \Omega_{\boldsymbol{X}} \times \mathbb{R}^+ &\longrightarrow \Omega_{\boldsymbol{x}} \times \mathbb{R}^+ \\ (\boldsymbol{X}, t) &\longmapsto \boldsymbol{\varphi}(\boldsymbol{X}, t) = (\boldsymbol{x}, t) \end{aligned}$$

and expresses the motion of the grid points in the reference domain. To be invertible and preserve the orientation, the Jacobian J of the mapping $\boldsymbol{\varphi}$ has to satisfy at each point and time the condition

$$\det\left(J(\boldsymbol{X}, t)\right) = \det\left(\frac{\partial \boldsymbol{x}(\boldsymbol{X}, t)}{\partial \boldsymbol{X}}\right) > 0\,. \tag{2.4}$$

The physical variables can be described differently according to the chosen configuration. Consider for instance a scalar physical quantity described by $f(\boldsymbol{x},t)$ in the spatial domain and by $F(\boldsymbol{X},t)$ in the reference one. The two functional forms are in general different, but their are related by $F(\boldsymbol{X},t) = f(\boldsymbol{\varphi}(\boldsymbol{X},t),t)$. A further remark has to be made also regarding the derivative operator. The time derivative in the ALE frame is defined as

$$\left.\frac{\partial f(\boldsymbol{x},t)}{\partial t}\right|_{\boldsymbol{X}} = \frac{\partial F(\boldsymbol{X},t)}{\partial t} \quad \text{with} \quad (\boldsymbol{X},t) = \boldsymbol{\varphi}^{-1}(\boldsymbol{x},t)\,.$$

Likewise, the grid velocity is defined as

$$\boldsymbol{v}(\boldsymbol{x},t) = \left.\frac{\partial \boldsymbol{x}}{\partial t}\right|_{\boldsymbol{X}}.$$

To obtain the ALE counterpart of the governing equations (2.1), the Eulerian time derivative $\left.\frac{\partial \mathsf{u}}{\partial t}\right|_{\boldsymbol{x}}$ has to be replaced by the time derivative in the ALE framework, which can be obtained by the chain rule as

$$\left.\frac{\partial \mathsf{u}}{\partial t}\right|_{\boldsymbol{X}} = \left.\frac{\partial \mathsf{u}}{\partial t}\right|_{\boldsymbol{x}} + \left.\frac{\partial \boldsymbol{x}}{\partial t}\right|_{\boldsymbol{x}} \cdot \boldsymbol{\nabla}_x \mathsf{u} = \left.\frac{\partial \mathsf{u}}{\partial t}\right|_{\boldsymbol{x}} + \boldsymbol{v} \cdot \boldsymbol{\nabla}_x \mathsf{u} \tag{2.5}$$

where $\boldsymbol{\nabla}_x(\,)$ indicates the gradient operator in Ω_x.

In order to establish the integral form of the conservation laws for mass, momentum, and energy, the Reynolds transport theorem is now exploited. As shown for instance in [13], the material time derivative of the integral of a scalar function $f(\boldsymbol{x},t)$ over a control volume $\mathcal{C}_x(t)$ bounded by a close surface $\partial\mathcal{C}_x$ moving with a velocity $\boldsymbol{v}(\boldsymbol{x},t)$ can be expressed as

$$\frac{d}{dt}\int_{\mathcal{C}_x(t)} f(\boldsymbol{x},t)\,d\boldsymbol{x} = \int_{\mathcal{C}_x(t)} \frac{\partial f(\boldsymbol{x},t)}{\partial t}\,d\boldsymbol{x} + \oint_{\partial\mathcal{C}_x(t)} f(\boldsymbol{x},t)\boldsymbol{v}\cdot\boldsymbol{n}d\boldsymbol{s}\,. \tag{2.6}$$

Applying the divergence theorem to the boundary integral, along with the Relations (2.5), one obtains

$$\begin{aligned}\frac{d}{dt}\int_{\mathcal{C}_x(t)} f(\boldsymbol{x},t)\,d\boldsymbol{x} &= \int_{\mathcal{C}_x(t)}\left[\left.\frac{\partial f}{\partial t}\right|_{\boldsymbol{x}} + \boldsymbol{\nabla}_x f\cdot\boldsymbol{v} + f\boldsymbol{\nabla}_x\cdot\boldsymbol{v}\right]d\boldsymbol{x} = \\ &= \int_{\mathcal{C}_x(t)}\left[\left.\frac{\partial f}{\partial t}\right|_{\boldsymbol{x}} + \boldsymbol{\nabla}_x\cdot(f\boldsymbol{v})\right]d\boldsymbol{x}\,.\end{aligned}$$

The Euler equations in the ALE framework for a control volume $\mathcal{C}(t) \in \Omega_x$ moving at velocity $\boldsymbol{v}$ are finally obtained as follows

$$\frac{d}{dt}\int_{\mathcal{C}(t)} \mathsf{u}\,d\boldsymbol{x} + \oint_{\partial\mathcal{C}(t)} [\mathbf{f}(\mathsf{u}) - \mathsf{u}\boldsymbol{v}]\cdot\boldsymbol{n}\,d\boldsymbol{s} = 0, \tag{2.7}$$

where, differently from Equations (2.1), $\boldsymbol{v}$ and $\boldsymbol{n}$ are also functions of time t. It should be pointed out that, with respect to the derivation of the ALE formulation here described, an inverse path is usually followed in practice, namely the grid velocity $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x},t)$ is generally known (prescribed or given by some computations) and then a compatible ALE mapping is built.

As recalled in Subsection 1.2.3, when solving the governing equations over moving domains, the fulfillment of the Geometric Conservation Law (GCL) is beneficial to avoid spurious oscillations and preserve non-linear stability [86, 44, 48, 98]. To this end, the basic requirement for an ALE scheme is the capability to exactly compute the trivial solution of a uniform flow. To translate this statement into a mathematically meaningful equation, the mass balance is usually selected among the ALE governing equations (2.7). Considering a uniform field over a moving volume $\mathcal{C}(t)$, it reduces to the following continuous geometric conservation law

$$\frac{d}{dt}\int_{\mathcal{C}(t)} d\boldsymbol{x} = \oint_{\partial\mathcal{C}(t)} \boldsymbol{v}\cdot\boldsymbol{n}\, d\boldsymbol{s}\,. \tag{2.8}$$

Integrating in time from t^n to $^{n+1}$ the previous equation leads to the Discrete Geometric Conservation Law (DGCL):

$$V^{n+1} - V^n = \int_{t^n}^{t^{n+1}} \left(\oint_{\partial\mathcal{C}(t)} \boldsymbol{v}\cdot\boldsymbol{n}\, d\boldsymbol{s} \right) dt\,, \tag{2.9}$$

where V^n and V^{n+1} are the volume of $\mathcal{C}$ at time t^n and t^{n+1} respectively. The DCGL states that the variation in volume in a certain time interval should balance the volume swept by its boundary during the same interval. This requirement represents a constraint that the grid velocity must fulfill during the motion.

2.2.1 Jacobian matrix and eigenstructure

The Jacobian matrix of the Euler flux function plays a crucial role in the development of the numerical scheme to solve the governing equations.

The Jacobian matrix A is defined as the partial derivatives of the flux function with respect to the solution vector projected along an arbitrary direction, identified for instance through the unit vector $\boldsymbol{n}$. In the ALE framework, it reads

$$\mathsf{A}^{\text{ALE}}(\mathsf{u},\boldsymbol{n},\boldsymbol{v}) = \frac{\partial \mathbf{f}(\mathsf{u})\cdot\boldsymbol{n}}{\partial \mathsf{u}} - (\boldsymbol{v}\cdot\boldsymbol{n})\mathbb{I}^5 = \mathsf{A}^{\text{EUL}}(\mathsf{u},\boldsymbol{n}) - (\boldsymbol{v}\cdot\boldsymbol{n})\mathbb{I}^5 \;\in\; \mathbb{R}^5\times\mathbb{R}^5\,. \tag{2.10}$$

The ALE Jacobian matrix is therefore the sum of the Jacobian matrix computed in the Eulerian formulation plus a correction term proportional to the grid velocity in the direction $\boldsymbol{n}$.

Thanks to the hyperbolic nature of the Euler equations, the Jacobian matrix can be expressed in terms of the matrices of the eigenvalues and eigenvectors by the following diagonal decomposition

$$\mathsf{A}^{\text{ALE}}(\mathsf{u},\boldsymbol{n},\boldsymbol{v}) = \mathsf{R}^{\text{ALE}}(\mathsf{u},\boldsymbol{v})\mathsf{\Lambda}^{\text{ALE}}(\mathsf{u},\boldsymbol{n},\boldsymbol{v})\mathsf{L}^{\text{ALE}}(\mathsf{u},\boldsymbol{v}) \tag{2.11}$$

where $\mathsf{\Lambda}^{\text{ALE}}(\mathsf{u},\boldsymbol{n},\boldsymbol{v})$ is the diagonal matrix of the eigenvalues and $\mathsf{R}^{\text{ALE}}(\mathsf{u},\boldsymbol{v})$ and $\mathsf{L}^{\text{ALE}}(\mathsf{u},\boldsymbol{v})$ are the matrices of the right and left eigenvectors, respectively [87].

The eigenvalues of the Euler equations within the ALE framework in three dimensions are

$$\mathsf{\Lambda}^{\text{ALE}}(\mathsf{u},\boldsymbol{n},\boldsymbol{v}) = \left(\frac{\boldsymbol{m}\cdot\boldsymbol{n}}{\rho} - \boldsymbol{v}\cdot\boldsymbol{n}\right)\mathbb{I}^5 - \operatorname{diag}\big(c(\mathsf{u}),\,0,\,0,\,0,\,-c(\mathsf{u})\big)$$

where c is the speed of sound. Similarly to the Jacobian matrix, also the ALE eigenvalues correspond to the ones in the Eulerian framework plus a term proportional to the projection of the grid velocity. On the contrary, since the last term

in equation (2.10) is already diagonal and it does not depend on the solution u, the eigenvectors matrices that diagonalize A^{ALE} and A^{EUL} are the same and they do not depend on the velocity $\boldsymbol{v}$. Therefore, in the following no distinction is made in the notation, i.e. $\mathsf{R}^{\text{ALE}} = \mathsf{R}^{\text{EUL}} = \mathsf{R}$ and $\mathsf{L}^{\text{ALE}} = \mathsf{L}^{\text{EUL}} = \mathsf{L}$.

A more detailed description of the eigenstructure of the Euler equations can be found in [113].

2.3 Node-pair finite volume discretization

The discrete counterpart of the Euler equations (2.7) over an unstructured dynamic grid with constant topology is obtained by means of the finite volume formulation. The computational domain $\Omega(t)$ is split into N_m non-overlapping volumes $\mathcal{C}_i(t)$, so that $\bigcup_i \mathcal{C}_i(t) \equiv \Omega(t)$. According to the node-centered approach considered here, the unknowns are associated to the nodes of the mesh and the control volume $\mathcal{C}_i$ that surrounds the node i is composed by a portion of all grid elements sharing the node i, as shown in Figure 2.2. For each finite volume, the Equations (2.7) becomes

$$\frac{d}{dt}\left[V_i \mathsf{u}_i\right] + \oint_{\partial\mathcal{C}_i(t)} \left[\mathbf{f}(\mathsf{u}) - \mathsf{u}\boldsymbol{v}\right] \cdot \boldsymbol{n}_i \, d\boldsymbol{s} = 0\,, \qquad \forall i \in \mathcal{K} \tag{2.12}$$

where

$$V_i(t) = \int_{\mathcal{C}_i(t)} d\boldsymbol{x} \qquad \text{and} \qquad \mathsf{u}_i(t) = \frac{1}{V_i(t)} \int_{\mathcal{C}_i(t)} \mathsf{u}(\boldsymbol{x}, t) \, d\boldsymbol{x}\,,$$

$\mathcal{K}$ is the set of all nodes of the triangulation and $\boldsymbol{n}_i$ is the outward unit normal to the boundary of the control volume $\mathcal{C}_i$. In the following, the variables of integration $d\boldsymbol{x}$ and $d\boldsymbol{s}$ are dropped for conciseness.

The flux integral in Equation (2.12) can be rearranged to put into evidence the domain and boundary contributions. Moreover, the domain contribution can be split into N_i contributions, where N_i is the number of the control volumes sharing a portion of their boundary with $\mathcal{C}_i$, $\mathcal{C}_i$ excluded. The set of this nodes is denoted by $\mathcal{K}_{i,\neq} = \{k \in \mathcal{K}, k \neq i \mid \partial\mathcal{C}_i \cap \partial\mathcal{C}_k \neq \emptyset\}$. The portion of the shared boundary $\partial\mathcal{C}_{ik}(t) = \partial\mathcal{C}_i \cap \partial\mathcal{C}_k$ is usually referred to as cell interface and it is associated to the couple of interacting nodes i and k, which is termed node-pair. Therefore, Equation (2.12) can be rewritten as

$$\frac{d}{dt}\left[V_i \mathsf{u}_i\right] + \sum_{k \in \mathcal{K}_{i,\neq}} \int_{\partial\mathcal{C}_{ik}(t)} \left[\mathbf{f}(\mathsf{u}) - \mathsf{u}\boldsymbol{v}\right] \cdot \boldsymbol{n}_i + \int_{\partial\mathcal{C}_i(t) \cap \partial\Omega(t)} \left[\mathbf{f}(\mathsf{u}) - \mathsf{u}\boldsymbol{v}\right] \cdot \boldsymbol{n}_i = 0 \tag{2.13}$$

where the last term—the contribution of the boundary fluxes—is different from zero only if a portion of $\partial\mathcal{C}_i$ lies on the boundary of the computational domain $\partial\Omega$, i.e. if $\partial\mathcal{C}_i(t) \cap \partial\Omega(t) \neq \emptyset$, see for instance the right picture in Figure 2.2.

A suitable approximation Φ of the integral of the fluxes across the cell interfaces is introduced and it will be referred to in the following as the integrated numerical flux. Considering for instance a centered approximation of the unknown and of the flux function at the cell interfaces, the domain contributions read

$$\begin{aligned} \int_{\partial\mathcal{C}_{ik}(t)} \left[\mathbf{f}(\mathsf{u}) - \mathsf{u}\boldsymbol{v}\right] \cdot \boldsymbol{n}_i \;&\simeq\; \frac{\mathbf{f}(\mathsf{u}_i) + \mathbf{f}(\mathsf{u}_k)}{2} \cdot \boldsymbol{\eta}_{ik} - \frac{\mathsf{u}_i + \mathsf{u}_k}{2} \nu_{ik} \\ &=\; \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}) \end{aligned} \tag{2.14}$$

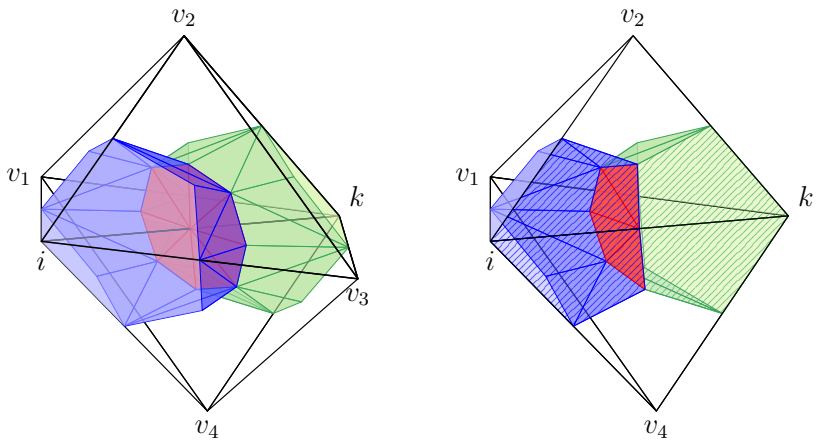

Figure 2.2: Portions of the finite volumes surrounding the nodes i and k associated to the four tetrahedral elements defined by the grid nodes i, k, v_1, v_2, v_3, v_4. In the left picture, domain (i.e. internal) grid elements are considered, while in the right one the faces i-v_2-k and i-k-v_4 are supposed lying on the boundary $\partial\Omega$. Grid edges are drawn by black lines. The portions of $\mathcal{C}_i$ and $\mathcal{C}_k$ are painted respectively blue and green, while their boundaries $\partial\mathcal{C}_i$ and $\partial\mathcal{C}_k$ are painted darker, i.e. and . The red color identifies the cell interface $\partial\mathcal{C}_{ik} = \partial\mathcal{C}_i \cap \partial\mathcal{C}_k$ associated to the node-pair i-k, which is shown entirely. For the boundary case (at right), the patterns and indicates the boundary interfaces $\partial\mathcal{C}_i \cap \partial\Omega$ and $\partial\mathcal{C}_k \cap \partial\Omega$.

where the metric quantities $\boldsymbol{\eta}_{ik}$ and ν_{ik} are, respectively, the integrated normal and interface velocity of $\partial\mathcal{C}_{ik}$, defined as

$$\boldsymbol{\eta}_{ik} = \int_{\partial\mathcal{C}_{ik}} \boldsymbol{n}_i \qquad \text{and} \qquad \nu_{ik} = \int_{\partial\mathcal{C}_{ik}} \boldsymbol{v} \cdot \boldsymbol{n}_i \,, \tag{2.15}$$

and where $\eta_{ik} = |\boldsymbol{\eta}_{ik}|$ and $\hat{\boldsymbol{\eta}}_{ik} = \boldsymbol{\eta}_{ik}/\eta_{ik}$.

A similar approximation Φ^{∂} is introduced also for the boundary contribution in Equation (2.13). Assuming that u_i is constant over the finite volume and therefore also along the boundary portion $\partial\mathcal{C}_i \cap \partial\Omega$, the boundary integral simplified to

$$\int_{\partial\mathcal{C}_i(t)\cap\partial\Omega(t)} [\mathbf{f}(\mathsf{u}) - \mathsf{u}\boldsymbol{v}] \cdot \boldsymbol{n}_i \;\simeq\; \mathbf{f}(\mathsf{u}_i) \cdot \boldsymbol{\xi}_i - \mathsf{u}_i \nu_i = \; \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i) \tag{2.16}$$

where, likewise the domain contribution, $\boldsymbol{\xi}_i$ and ν_i are the integrated normal and the interface velocity of the boundary interface associated to the node i

$$\boldsymbol{\xi}_i = \int_{\partial\mathcal{C}_i(t)\cap\partial\Omega(t)} \boldsymbol{n}_i \qquad \text{and} \qquad \nu_i = \int_{\partial\mathcal{C}_i(t)\cap\partial\Omega(t)} \boldsymbol{v} \cdot \boldsymbol{n}_i \,, \tag{2.17}$$

and where $\xi_i = |\boldsymbol{\xi}_i|$ and $\hat{\boldsymbol{\xi}}_i = \boldsymbol{\xi}_i/\xi_i$.

To guarantee that the scheme is conservative and that the finite volumes are closed, the metric vectors introduced in Equations (2.14) and (2.16) must satisfy

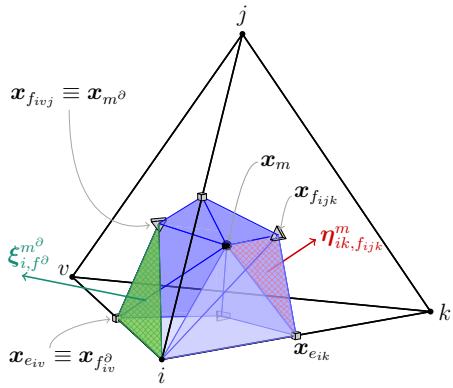

Figure 2.3: Domain and boundary metrics definition. The six triangular facets defining the contribution of the considered element m to the finite volume $\mathcal{C}_i$ are depicted in blue ■. Each one has as vertices the barycenter of the element ●, the barycenter of a face and that one of an edge. The red pattern highlights the triangle related to the edge e_{ik} and the face i-j-k, over which the elemental contribution to the integrated normal $\boldsymbol{\eta}_{ik}$ shown by the red arrow is computed. Moreover, supposing that the face i-v-j lies on the boundary, the contribution to the boundary metric vector $\boldsymbol{\xi}^{m^\partial}_{i,f^\partial}$ is shown, where the boundary element m^∂ consists in the triangle with vertices i-v-j and the considered face f^∂ is the edge i-v. The green pattern depicts the boundary elemental portion $\partial\mathcal{C}^{m^\partial}_{i,f^\partial}$. Finally, the volume $V^m_{ik,f}$ of the sub-tetrahedron that has $\partial\mathcal{C}^m_{ik,f}$ as base and the node i as opposite vertex is shown in light blue ■.

the following conditions [116]:

$$\boldsymbol{\eta}_{ik} = -\boldsymbol{\eta}_{ki} \tag{2.18}$$

$$\nu_{ik} = -\nu_{ki} \tag{2.19}$$

$$\boldsymbol{\xi}_i(t) + \sum_{k \in \mathcal{K}_{i,\neq}} \boldsymbol{\eta}_{ik}(t) = \mathbf{0}\,. \tag{2.20}$$

Finally, the node-pair representation of the spatially discrete governing equations (2.12) reads

$$\frac{d}{dt}\left[V_i \mathsf{u}_i\right] = -\sum_{k \in \mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}) \; - \; \Phi^\partial(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i) \qquad \forall i \in \mathcal{K} \tag{2.21}$$

which consists in $5 \times N_V$ Ordinary Differential Equations (ODE), where N_V is the number of grid points.

2.3.1 Metrics computation for a tetrahedral mesh

In this section, the expression for the metric quantities $\boldsymbol{\eta}_{ik}$ and $\boldsymbol{\xi}_i$ and the volumes V_i are given in terms of grid nodes position. The time dependence of the position of the grid points, and therefore of all geometrical quantities involved in metric computation, is here omitted to lighten the notation. Since the three-dimensional adaptive scheme described in the following chapter can be applied, at least at the current level of development, only to tetrahedral grids, only this kind of elements

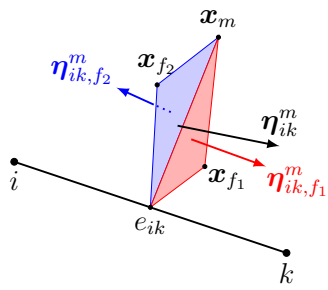

Figure 2.4: Orientation of the elemental contribution to the integrated normal. The global orientation of the edge e_{ik} is from i to k and it is shared by the faces f_1 and f_2 of the element m. The picture shows the elemental interface contribution $\partial\mathcal{C}^m_{ik,f}$ for both faces. Suppose that for the face f_1, the edge e_{ik} has the local orientation equal to the global one, while for the face f_2 it is oriented in the opposite way. Then, when computing $\boldsymbol{\eta}^m_{ik}$, the contribution $\boldsymbol{\eta}^m_{ik,f_1}$ is summed with the positive sign while $\boldsymbol{\eta}^m_{ik,f_2}$ is taken with the negative sign.

are considered in the present description, even if the described approach can be applied to different types of grid elements.

The control volumes over which the governing equations are enforced are built according to the median-dual mesh approach [11, 131]. Therefore, each control volume $\mathcal{C}_i$ is bounded by the median planes of the tetrahedra having node i as a vertex. Conversely, each grid tetrahedron contributes to the control volumes surrounding its four nodes. With reference to Figure 2.3, the description of the cell interface $\partial\mathcal{C}_{ik}$ between the finite volumes $\mathcal{C}_i$ and $\mathcal{C}_k$ is given.

First of all, it is to be noticed that the generic interface $\partial\mathcal{C}_{ik}$ is associated to the grid edge e_{ik} between the nodes i and k, therefore each element m that includes the edge e_{ik} contributes to the interface with its elemental interface contribution $\partial\mathcal{C}^m_{ik}$. This elemental interface is composed by two triangles $\partial\mathcal{C}^m_{ik,f}$ relating to the different faces f of the tetrahedron m to which the edge e_{ik} belongs. The three nodes of the triangle are the barycenters of the element $\boldsymbol{x}_m$, of the edge $\boldsymbol{x}_{ik}$ and of the face $\boldsymbol{x}_f$. According to this partition of the interface, the integrated normals are computed on each triangular portion $\partial\mathcal{C}^m_{ik,f}$ and then summed together.

The elemental contribution to the integrated normal $\boldsymbol{\eta}_{ik}$ of the element m for the face f can be computed as

$$\boldsymbol{\eta}^m_{ik,f} = -\frac{1}{2}\left(\boldsymbol{x}_f - \boldsymbol{x}_m\right) \times \left(\boldsymbol{x}_{ik} - \boldsymbol{x}_m\right) . \tag{2.22}$$

where the minus sign makes the normal oriented as the node-pair c_{ik}, which is define so that $i < k$.

Similarly, the boundary metric vector $\boldsymbol{\xi}_i$ is built by summing the contributions of all boundary elements sharing the nodes i. The elemental contribution $\boldsymbol{\xi}^{m^\partial}_{i,f^\partial}$ is a triangle formed by the boundary node i, the barycenters of the element m^∂ and of the face f^∂ and it reads

$$\boldsymbol{\xi}^{m^\partial}_{i,f^\partial} = -\frac{1}{2}\left(\boldsymbol{x}_{f^\partial} - \boldsymbol{x}_{m^\partial}\right) \times \left(\boldsymbol{x}_i - \boldsymbol{x}_{f^\partial}\right) \tag{2.23}$$

Note that for a tetrahedral grid, the boundary elements are triangles, therefore the faces of the boundary element correspond to the edges of the triangle.

The metric vectors $\boldsymbol{\eta}_{ik}$ and $\boldsymbol{\xi}_i$ can be then computed by summing all the elemental contributions

$$\boldsymbol{\eta}_{ik} = \sum_{m \in \varepsilon_i \cap \varepsilon_k} \sum_{f \in \mathcal{F}_{ik,m}} \boldsymbol{\eta}^m_{ik,f} \qquad \text{and} \qquad \boldsymbol{\xi}_i = \sum_{m^\partial \in \varepsilon_i^\partial} \sum_{f \in \mathcal{F}_{i,m^\partial}} \boldsymbol{\xi}^{m^\partial}_{i,f^\partial}, \tag{2.24}$$

where ϵ_i and ϵ_k are the set of the elements sharing the node i and k, respectively (often referred to as the element bubble of i and k), $\mathcal{F}_{ik,m}$ is set of faces of m sharing the edge ik, ε_i^∂ is the set of boundary elements sharing the nodes i and $\mathcal{F}_{i,m^\partial}$ is the set of faces of m^∂ sharing i. The set $\varepsilon_i \cap \varepsilon_k$ denotes the set of all elements sharing the edge ik.

When summing all contributions, particular care must be taken to preserve the correct orientation of the metric vectors. Each edge of the tetrahedron has an elemental orientation and a local one, defined so that the normal of each face points inside the element, i.e. the edges are oriented clockwise on each face. The minus sign in Equations (2.22) makes the contribution $\boldsymbol{\eta}^m_{ik,f}$ oriented as the local edge ik of the face f, as Figure 2.4 shows. When performing the internal sum in Equation (2.24), the elemental contribution $\boldsymbol{\eta}^m_{ik,f}$ is inverted if the global and local orientations of the edge e_{ik} are opposite for the face f. Moreover, when summing the contribution of different elements, a further inversion of the elemental contribution is required if the element edge is not oriented as the node-pair, which is defined as going from i to k with $i < k$. On the contrary, the boundary elemental contributions are computed always as oriented outwards the domain. This can be accomplished by inverting the elemental contribution $\boldsymbol{\xi}^{m^\partial}_{i,f^\partial}$ if i is the second node of the edge e_{ik} (face for the boundary element).

In a similar way, also the volume V_i of the finite volume $\mathcal{C}_i$ is computed by summing elemental contribution. According to the subdivision described above, each tetrahedron is split into 24 sub-tetrahedra, each of them has the triangular interface $\partial\mathcal{C}^m_{ik,f}$ as base and the node i or k as the opposite vertex. The volume of the sub-tetrahedron with vertices $\boldsymbol{x}_i$, $\boldsymbol{x}_m$, $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_f$ is

$$V^m_{ik,f} = \frac{1}{3} \left(\boldsymbol{x}_m \times \boldsymbol{x}_i \right) \cdot \boldsymbol{\eta}_{ik,f} \,. \tag{2.25}$$

It can be noticed that if the vertex $\boldsymbol{x}_k$ is used instead of $\boldsymbol{x}_i$, the volume $V^m_{ik,f}$ is equal with opposite sign. Provided that the correct orientation of the contributions is obtained according to the similar considerations made for the integrated normal $\boldsymbol{\eta}_{ik}$, the volume of the finite volume $\mathcal{C}_i$ can be computed as

$$V_i = \sum_{m \in \varepsilon_i} \sum_{k \in \mathcal{K}^m_{i,\neq}} \sum_{f \in \mathcal{F}_{ik,m}} V^m_{ik,f} \tag{2.26}$$

where $\mathcal{K}^m_{i,\neq}$ are the three vertices of the tetrahedron m different from i.

The proposed node-pair finite volumes discretization is straightforward to implement. Thanks to the grid transparency [67], after the median-dual mesh is generated, all information about elements can be discarded and an edge-based data structure that makes no distinction between 2D or 3D or different element types can be used within the flow solver. Moreover, Selmin [115] and Selmin and Formaggia [116] proved that for fixed simplex grids (triangular in 2D and

tetrahedral in 3D) the described finite volume discretization is equivalent to a finite element one with linear elements, except a different treatment of boundary terms.

2.3.2 Integrated numerical fluxes

When evaluating the numerical fluxes in the spatially discretized equations two different aims are pursued: to correctly capture the possible discontinuities in the solution and to achieve second-order spatial accuracy. To this end, in this work a high-resolution scheme based on the Total Variation Diminishing (TVD) approach is used for integrated numerical fluxes [123]. Thus, a first order monotonicity preserving scheme is blended with a second-order scheme by means of a limiter, as briefly described in the following. A more detailed explanation can be found for instance in [87].

At each cell interface, the domain fluxes in Equation (2.21) are calculated by the following second-order centered approximation

$$\Phi^{\mathsf{II}}(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}) = \frac{\mathbf{f}(\mathsf{u}_i) + \mathbf{f}(\mathsf{u}_k)}{2} \cdot \boldsymbol{\eta}_{ik} - \frac{\mathsf{u}_i + \mathsf{u}_k}{2} \nu_{ik} \,. \tag{2.27}$$

It is well known that the previous definition of the integrated numerical fluxes leads to undesirable spurious oscillations near discontinuities [87]. Therefore, where the solution is discontinuous the approximate Riemann solver proposed by Roe is employed, which solves at each cell interface a linear substitute of the classical Riemann problem of gas dynamics, taking the nodal values of the solution as left and right states [112]. Then, the first-order upwind flux reads

$$\Phi^{\mathsf{I}}(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}) = \frac{\mathbf{f}(\mathsf{u}_i) + \mathbf{f}(\mathsf{u}_k)}{2} \cdot \boldsymbol{\eta}_{ik} - \frac{\mathsf{u}_i + \mathsf{u}_k}{2} \nu_{ik} - \frac{1}{2} |\widetilde{\mathsf{A}}| (\mathsf{u}_k - \mathsf{u}_i) \tag{2.28}$$

where $\widetilde{\mathsf{A}}$ is the Roe matrix, defined as the Jacobian of the flux function projected along the normal direction $\hat{\boldsymbol{\eta}}_{ik}$ and evaluated at an intermediate state $\widetilde{\mathsf{u}} = \widetilde{\mathsf{u}}(\mathsf{u}_i, \mathsf{u}_k)$, namely $\widetilde{\mathsf{A}} = \mathsf{A}^{\mathrm{ALE}}(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}, \nu_{ik})$. If the polytropic ideal gas model is assumed, the intermediate state can be computed as

$$\widetilde{\boldsymbol{m}} = \frac{\boldsymbol{m}_i \sqrt{\rho_k} + \boldsymbol{m}_k \sqrt{\rho_i}}{\sqrt{\rho_i} + \sqrt{\rho_k}} \,, \qquad \widetilde{h^{\mathrm{t}}} = \frac{h^{\mathrm{t}}_i \sqrt{\rho_k} + h^{\mathrm{t}}_k \sqrt{\rho_i}}{\sqrt{\rho_i} + \sqrt{\rho_k}} \,,$$

where $(h^{\mathrm{t}} = E^{\mathrm{t}} + P)\rho$ is the specify total enthalpy. The intermediate density is arbitrarily chosen as $\widetilde{\rho} = \sqrt{\rho_i \rho_k}$, since for an ideal polytropic gas the Roe matrix is independent from its value. The extension of the Roe linearization to the van Der Waals Gas is derived by Guardone and Vigevano in [62].

To compute the absolute value of the Roe matrix, the latter is factorized by a spectral decomposition, i.e. $\widetilde{\mathsf{A}} = \widetilde{\mathsf{R}} |\widetilde{\Lambda}| \widetilde{\mathsf{L}}$, where $\widetilde{\mathsf{R}} = \mathsf{R}(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}_{ik})$ and $\widetilde{\mathsf{L}} = \mathsf{L}(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}_{ik})$ are the right and left eigenvectors matrices introduced in Equation (2.11) evaluated in the Roe state. The diagonal matrix $|\widetilde{\Lambda}|$ is built by the absolute values of the eigenvalues of $\widetilde{\mathsf{A}}$ integrated over the interface $\partial \mathcal{C}_{ik}$, namely

$$\big|\Lambda(\widetilde{\mathsf{u}}, \boldsymbol{\eta}_{ik}, \nu_{ik})\big| = \left(\frac{\widetilde{\boldsymbol{m}} \cdot \boldsymbol{\eta}_{ik}}{\widetilde{\rho}} - \nu_{ik} \right) \mathbb{I}^5 - \mathrm{diag}\big(c(\widetilde{\mathsf{u}}),\, 0,\, 0,\, 0,\, -c(\widetilde{\mathsf{u}})\big) \,.$$

As a final remark, it should be noted that the upwind term in (2.28) can be interpreted as a dissipation term. When numerical dissipation is too small, the Roe linearization may lead to non-physical solutions that do not respect the entropy condition, i.e. the additional constraint to the solution scheme that guarantees that the conservation laws converge to the unique physically relevant, physical weak solution [87]. This drawback occurs when the eigenvalues of the Roe matrix $\widetilde{\lambda}$ are close to zero, therefore it can be avoided by a suitable modification of the eigenvalues, which is often called entropy fix [106]. In the present work, the modification of the entropy fix proposed by Selmin [115] is employed and the p-th eigenvalue of the Roe matrix is modified as follows

$$\widehat{\lambda}_p = \begin{cases} |\widetilde{\lambda}_p| & \text{if } \widetilde{\lambda}_p > \widetilde{\delta}, \\ \dfrac{\widetilde{\lambda}_p^2 + \widetilde{\delta}^2}{2\widetilde{\delta} + \epsilon} & \text{if } \widetilde{\lambda}_p < \widetilde{\delta}, \end{cases} \tag{2.29}$$

where $\epsilon = 10^{-12}$ is a small parameter to avoid divisions by zero, and

$$\widetilde{\delta} = \frac{1}{5}\big(\mathrm{M}(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}, \nu_{ik}) + 1\big)\eta_{ik} \quad \text{and} \quad \mathrm{M}\,(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}, \eta, \nu) = \frac{1}{c(\widetilde{\mathsf{u}})}\left(\frac{\widetilde{\boldsymbol{m}} \cdot \hat{\boldsymbol{\eta}}}{\widetilde{\rho}} - \frac{\nu}{\eta}\right)$$

where M in the above relations is the Mach number of the intermediate state projected along the $\hat{\boldsymbol{\eta}}_{ik}$ direction.

Flux limiter

The Total Variation Diminishing approach [66] is used to obtained the high-resolution expression for the integrated numerical domain fluxes. A suitable flux limiter is used to control the switch from the centered second-order approximation Φ^{II} to the monotonicity preserving scheme Φ^{I} near discontinuities. Denoting the limiter by $\Upsilon = \mathrm{diag}(\Upsilon_1, ...\Upsilon_5)$, the high resolution integrated flux reads

$$\Phi^{\mathsf{HR}}_{ik} = \Phi^{\mathsf{I}}_{ik} + \Upsilon(\Phi^{\mathsf{II}}_{ik} - \Phi^{\mathsf{I}}_{ik}) = \Phi^{\mathsf{II}}_{ik} + \frac{1}{2}\widetilde{\mathsf{R}}|\Lambda|\left[\Upsilon - \mathbb{I}^5\right]\widetilde{\mathsf{L}}(\mathsf{u}_k - \mathsf{u}_i) \tag{2.30}$$

with Υ defined according to the expression of the van Leer limiter [123]. The limiters are usually defined in terms of the ratio between the variation of the solution in the upwind and the centered scheme. To evaluate the upwind jump over an unstructured mesh, it is convenient to recast to an extended node-pair structure. Following [61], for each node-pair i-k, two additional nodes $i^\star$ and $k^\star$ are considered, that are selected as the best aligned node-pair with direction i-k as shown in Figure 2.5. Thus, the p-th component of the characteristic upwind jump–i.e. multiplied by the corresponding eigenvector—can be computed as

$$\widetilde{\mathsf{q}}_p = \begin{cases} \dfrac{\hat{\boldsymbol{\eta}}_{ik} \cdot (\boldsymbol{x}_k - \boldsymbol{x}_i)}{\hat{\boldsymbol{\eta}}_{ik} \cdot (\boldsymbol{x}_{k^\star} - \boldsymbol{x}_k)}\,\widetilde{\mathsf{L}}_p(\mathsf{u}_{k^\star} - \mathsf{u}_k) & \text{if } \widetilde{\lambda}_p > 0\,, \\ \dfrac{\hat{\boldsymbol{\eta}}_{ik} \cdot (\boldsymbol{x}_k - \boldsymbol{x}_i)}{\hat{\boldsymbol{\eta}}_{ik} \cdot (\boldsymbol{x}_i - \boldsymbol{x}_{i^\star})}\,\widetilde{\mathsf{L}}_p(\mathsf{u}_i - \mathsf{u}_{i^\star}) & \text{if } \widetilde{\lambda}_p \leq 0\,, \end{cases} \tag{2.31}$$

where $\widetilde{\mathsf{L}}_p$ is the p-th row of the left eigenvector matrix.

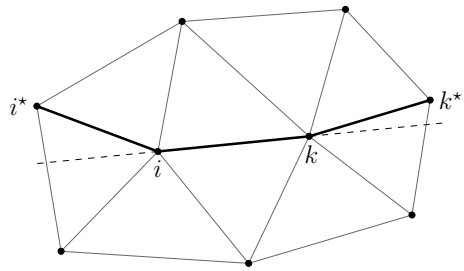

Figure 2.5: Extended node-pair structure for the node-pair i-k for the high-resolution integrated flux. For the sake of clarity, a bi-dimensional grid is sketched.

Boundary fluxes

The boundary conditions are enforced in a weak form, i.e. the imposed boundary values are taken into account by evaluating the flux Φ_i^∂ in a suitable boundary state $\mathsf{u}^\partial = \mathsf{u}^\partial(\mathsf{u}_i, \hat{\boldsymbol{\xi}}_i, \xi_i, \nu_i)$. Two types of boundary conditions are used here: slip (or symmetry) and non-reflecting.

Slip boundary conditions are imposed by subtracting to the state u_i the normal component of the velocity, taking into account mesh velocity, namely

$$\mathsf{u}_i^{\partial,\mathrm{W}} = \mathsf{u}_i - \left[0,\, \left(\boldsymbol{m}_i \cdot \hat{\boldsymbol{\xi}}_i - \frac{\rho_i \nu_i}{\xi_i}\right) \hat{\boldsymbol{\xi}}_i,\, \frac{1}{2}\rho_i \left| \frac{\boldsymbol{m}_i \cdot \hat{\boldsymbol{\xi}}_i}{\rho_i} - \frac{\nu_i}{\hat{\boldsymbol{\xi}}_i} \right|^2 \right]^{\mathrm{T}}.$$

Thus, the corresponding boundary flux simplifies to

$$\Phi_i^\partial(\mathsf{u}_i^{\partial,\mathrm{W}}, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i) = \Pi(\mathsf{u}_i) \left[0,\, \xi_i \hat{\boldsymbol{\xi}}_i,\, \nu_i\right]^{\mathrm{T}},$$

which corresponds to pressure forces and their power only.

As regards the enforcement of the non-reflecting boundary condition, due to the hyperbolic nature of the Euler equations, the same portion of a boundary can be both an inflow and an outflow boundary for different characteristic field. Therefore, the boundary state $\mathsf{u}_i^{\partial,\infty}$ is computed via characteristic reconstruction [61] as follows

$$\mathsf{u}_i^{\partial,\infty}(\mathsf{u}_i, \hat{\boldsymbol{\xi}}_i, \xi_i, \nu_i) = \mathsf{u}_i + \mathsf{R}(\mathsf{u}_i, \hat{\boldsymbol{\xi}}_i)\ \mathcal{SN}(\mathsf{u}_i, \hat{\boldsymbol{\xi}}_i, \xi_i, \nu_i)\ \mathsf{L}(\mathsf{u}_i, \hat{\boldsymbol{\xi}}_i)\,[\mathsf{u}_\infty - \mathsf{u}_i]\,,$$

where $\mathcal{SN}$ is an operator that selects only the strictly negative eigenvalues and stores them in a diagonal matrix. Thus, for a supersonic outflow the above relation simplifies to $\mathsf{u}_i^{\partial,\infty} = \mathsf{u}_i$, since all eigenvalues are negative, and to $\mathsf{u}_i^{\partial,\infty} = \mathsf{u}_\infty$ for a supersonic inflow, when all eigenvalues are positive.

2.4 Time discretization

The finite volume discretization of governing equations are now discretized in time by means of a Backward Differences Formula (BDF). According to the

approximation of order $p+1$, the derivative with respect to the time of a function f can be expressed as

$$\frac{dy}{dt} \simeq \frac{1}{\Delta t}\sum_{q=-1}^{p} a_q y^{n-q} \quad \text{or, equivalently} \quad \frac{dy}{dt} \simeq \frac{1}{\Delta t}\sum_{q=-1}^{p-1} \alpha_q \Delta y^{n-q}$$

where a_q are the coefficient of the BDF scheme [82], which can be expressed as $\alpha_q = \sum_{d=-1}^{q} a_q$ to obtain the incremental form where $\Delta y^n = y^n - y^{n-1}$ by exploiting the property $\sum_{q=-1}^{p} a_q = 0$. The definitions above applied to the ODE system (2.21) give

$$\begin{aligned}\sum_{q=-1}^{p} a_q V_i^{n-q} \mathsf{u}_i^{n-q} = \Delta t \Bigg[& \sum_{k\in\mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}) \\ & + \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i)\Bigg]^{n+1}, \qquad i \in \mathcal{K}\end{aligned} \tag{2.32}$$

where $\Phi(\cdot)^{n+1} = \Phi(\cdot^{n+1})$. In the previous system, all quantities at time level n are known, as well as all the grid-dependent quantities (cell volumes and metric vectors) at time level $n+1$, since the position of the grid points is known as a function of time. It is important to notice that this does not occur in fluid-structure interaction problems, in which the node velocity is the outcome of the solution of the structural problem with aerodynamic loads.

The System (2.32) is non-linear for the unknown variable u_i at time level $n+1$, therefore an iterative procedure is used to compute the solution. As it is standard practice, the convergence towards the solution is enhanced by resorting to a pseudo-time stepping scheme [72]. Moreover, since the second-order spatial accuracy and the mixed elliptic-hyperbolic nature of the equations result in a poorly diagonal dominant matrix, a modified version of the Newton method is used to solver the linear system. According to the defect-correction method [78], the exact Jacobian of the integrated flux function is replaced by an approximated one. At each psuedo-time step, a Simmetric Guass Seidel method is used to obtain the solution of the system of linear equations.

2.4.1 Iterative implicit solver

The iterative procedure used to solve the System (2.32) is now described. The equations for all nodes can be rewritten in a more compact form as

$$\frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1}\mathbf{U}^{n+1} + \mathbf{S} + \mathbf{R}(\mathbf{U}^{n+1}) = 0 \tag{2.33}$$

where $\mathbf{U} = [\mathsf{u}_1, \mathsf{u}_2, ..., \mathsf{u}_{N_V}]^{\mathrm{T}}$ is the vector of solutions, the diagonal matrix $\mathbf{V} = \mathrm{diag}(V_1\mathbb{I}^5, ..., V_{N_V}\mathbb{I}^5)$ contains the cell volumes, $\mathbf{S} = \sum_{q=0}^{p} a_q \mathbf{V}^{n-q}\mathbf{U}^{n-q}$ is treated as a source term and $\mathbf{R}$ is the residual vector computed as the right hand side, i.e. its i-th row is $\mathsf{R}_i = -\sum_{k\in\mathcal{K}_{i,\neq}} \Phi_{ik}^{n+1} - \Phi_i^{\partial,n+1}$.

All the terms of the above equation can be gathered in a so-called unsteady residual $\mathbf{R}^{\star}(\mathbf{U}^{n+1})$ that should be driven to zero at each time step, namely

$$\mathbf{R}^{\star}(\mathbf{U}^{n+1}) = \frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1}\mathbf{U}^{n+1} + \mathbf{S} + \mathbf{R}(\mathbf{U}^{n+1}) = 0\,. \tag{2.34}$$

The equation system above can be solved, in principle, by a Newton-Raphson scheme. However, in order to converge such iterative methods require that the initial solution is close enough to the final one. Unfortunately, this is not always the case if the governing equations are highly non-linear. To overcome this possible drawback, a pseudo time derivative is included in Equation (2.34) and an iterative procedure is used to obtain the solution at time level $n+1$ [72]. In the following the pseduo time is denoted τ and the time levels m and $m+1$ refer to pseudo-time iterations. Thus, knowing the solution $\mathbf{U}^n$, the solution $\mathbf{U}^{n+1}$ is obtained by solving the following equation

$$\frac{d\mathbf{U}}{d\tau} + \mathbf{R}^\star(\mathbf{U}^{m+1}) = 0\,, \tag{2.35}$$

where $\mathbf{U}^{m=0} = \mathbf{U}^n$ and convergence is achieved when

$$\|\mathbf{R}^\star(\mathbf{U}^{m+1})\|_2 \leq \mathsf{tol}\,\|\mathbf{R}^\star(\mathbf{U}^0)\|_2\,,$$

where tol is a user-defined tolerance, usually chosen in the range $10^{-8} \leq \mathsf{tol} \leq 10^{-5}$ depending on the application. At this point, the solution for the new time step is obtained, namely $\mathbf{U}^{n+1} = \mathbf{U}^{m+1}$.

The pseudo-time derivative in Equation (2.35) is discretized by means of the backward Euler scheme and the unsteady residual can be approximated by a first-order Taylor expansion to obtain

$$\left(\frac{1}{\Delta\tau} + \frac{\partial\mathbf{R}^\star}{\partial\mathbf{U}}\right)(\mathbf{U}^{m+1} - \mathbf{U}^m) + \mathbf{R}^\star(\mathbf{U}^m) = 0\,.$$

Expanding the definition of the unsteady residual at τ^m and its derivative, the above equation reads

$$\left(\frac{1}{\Delta\tau} + \frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1} + \frac{\partial\mathbf{R}}{\partial\mathbf{U}}\right)(\mathbf{U}^{m+1} - \mathbf{U}^m) = -\frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1}\mathbf{U}^m - \sum_{q=0}^{p} a_q \mathbf{V}^{n-q}\mathbf{U}^{n-q} - \mathbf{R}(\mathbf{U}^m) \tag{2.36}$$

which if convergence is obtained reduces to Equation (2.33).

At each pseudo step, the linear System (2.36) has to be solved. To this end, it can be re-written as $\mathbf{M}\mathbf{z} = \mathbf{q}^m$. As already said, the matrix $\mathbf{M}$ is poorly diagonally dominant. This occurs especially for steady problems, where the term $a_{-1}/\Delta t$ is not present. According to defect-correction approach [78], the Jacobian of $\mathbf{R}(\mathbf{U})$ is replaced by a first order approximation, $\widetilde{\mathbf{R}}(\mathbf{U})$, to increase its diagonally dominance. More specifically, only the first-order upwind flux in Equation (2.30) is considered when computing the Jacobian $\widetilde{\mathbf{R}}(\mathbf{U})$. This modification increases the robustness of the scheme, but unfortunately the quadratic convergence rate, which could be potentially achieved by the exact Newton method, is lost [125, 96].

Due to the non-linear nature of Equation (2.36), both the matrix $\mathbf{M}$ and the residual vector $\mathbf{q}^m$ change every pseudo-time step, namely every non-linear iteration. Therefore, as it is common practice, $\mathbf{U}^{m+1}$ is computed by means of an iterative scheme, here the Symmetric Guass Siedel, performing only a few number of so-called linear iterations without seeking the full convergence [125, 96]. Thus,

Test case	Reference	γ	Co^m	$\mathrm{Co}^{\max}$
Steady 3D	M6 wing (Section 6.1)	0.01	0.9	1.2
Steady 3D	Infinite-span NACA 0012 (Subsection 6.2.1)	0.02	0.9	5.0
Unsteady 2D	Airfoil NACA 0012 (Subsection 5.1.2)	0.05	0.95	$1.0\ 10^5$
Unsteady 3D	Infinite-span NACA 0012 (Subsection 6.2.2)	0.01	0.9	$2.5\ 10^5$

Table 2.2: Example values for the parameters used to update the Courant number during pseudo-time iterations.

the iterative scheme is stopped when the ratio between the norm of the linear residual ($\mathbf{q}_{\mathsf{lin}}^k = \mathbf{q}^m - \mathbf{M}\mathbf{z}^k$) and the one of the non-linear residual ($\mathbf{q}^m$) goes below a certain threshold, for instance when $\|\mathbf{q}_{\mathsf{lin}}^k\|_2 \leq 0.1\|\mathbf{q}^m\|_2$. A very low value of this threshold is not useful since the linear system is approximate and reaching a highly precise solution does not lead to a considerable improvement of the overall (non-linear) convergence [21].

More details about the implicit iterative solution of the linear system, and in particular about the construction of its matrix, can be found in [21, 70].

Local time stepping

A local time stepping technique is used when solving System (2.36). The local pseudo-time step $\Delta\tau_i$ is defined as

$$\Delta\tau_i = \frac{V_i \mathrm{Co}}{\sum_{k \in \mathcal{K}_{i,\neq}} \lambda_{\max}(\mathsf{u}_i, \mathsf{u}_k, \boldsymbol{\eta}_{ik}, \nu_{ik})}$$

where Co is the Courant number and $\lambda_{\max}$ is the eigenvalue of the Roe matrix with the largest absolute value, as proposed by Giles [57]. The Courant-Friedrichs-Lewy condition states that a necessary condition for the stability of the linear hyperbolic equation is that the Courant number has to be lower than or equal to 1 [87]. However, with the solution $\mathbf{U}^{m+1}$ approaching $\mathbf{U}^{n+1}$, i.e. when the residual becomes smaller and smaller, the Courant number can be increased to speed up the convergence, because the backward Euler integration scheme is unconditionally stable for linear system. Therefore, at each pseudo-time step the Courant number is modified as

$$\mathrm{Co}^{m+1} = \min\left(\max\left[\gamma \frac{\|\mathbf{R}(\mathbf{U}^{m-1})\|_2}{\|\mathbf{R}(\mathbf{U}^m)\|_2}, 1\right] \mathrm{Co}^m, \mathrm{Co}^{\max}\right)$$

where $\gamma = 0.8 \div 1$ is a user-defined parameter to control the increase, $\|\cdot\|_2$ denotes the L_2 norm and $\mathrm{Co}^{\max}$ is the maximum allowed value, introduced to avoid the Courant number to grow indefinitely.

In steady simulations, a uniform initial conditions equal to the free-stream state is used as initial guess, so the initial solution is quite far from the final one. This fact, along with the absence of the term $a_{-1}/\Delta t$ in Equation (2.36) that reduces the diagonally dominance of the matrix, imposes a small value of the parameter γ and of the maximum Courant number. For instance, typical values of the parameters used in the numerical simulations presented in Chapters 5 and 6 are reported in Table 2.2.

2.5 The IVC condition and the DGCL

The same finite volume discretization applied to the governing equations is used also to obtain the spatially discrete counterpart of the Geometric Conservation Law (2.8), namely

$$\frac{dV_i(t)}{dt} = \int_{\partial\mathcal{C}_i(t)} \boldsymbol{v}\cdot\boldsymbol{n}_i = \sum_{k\in\mathcal{K}_{i,\neq}} \nu_{ik}(t) + \nu_i(t)\,. \tag{2.37}$$

Following the geometrical partition used to compute the metric vectors, see Equations (2.24), the portion of the cell volume V_i related to the interface $\partial\mathcal{C}_{ik}$ can be defined as

$$V_{ik} = \sum_{m\in\varepsilon_i\cap\varepsilon_k} \sum_{f\in\mathcal{F}_{ik,m}} V^m_{ik,f}$$

where $V^m_{ik,f}$ is the volume of the sub-tetrahedron with vertices $\boldsymbol{x}_i$, $\boldsymbol{x}_m$, $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_f$ computed through Equation (2.25). Therefore, Equation (2.37) can be automatically satisfied by splitting the derivative of the cell volume into contributions pertaining to the different involved interfaces. It should be noticed that, even no portion of the volume cell is associated to the boundary interface $\partial\mathcal{C}_i(t)\cap\partial\Omega(t)$, its contribution to the derivative $\frac{dV_i}{dt}$ is not null. The so-called Interface Velocity Consistency (IVC) conditions [102] are thus obtained

$$\frac{dV_{ik}(t)}{dt} = \nu_{ik}(t)\,, \quad \forall k\in\mathcal{K}_{i,\neq} \qquad \text{and} \qquad \frac{dV_{i,\partial}(t)}{dt} = \nu_i(t)\,, \tag{2.38}$$

where $\frac{dV_{ik}}{dt}$ and $\frac{dV_{i,\partial}}{dt}$ are the volume swept by the interfaces $\partial\mathcal{C}_{ik}(t)$ and $\partial\mathcal{C}_i(t)\cap\partial\Omega(t)$, respectively.

If the grid point displacement is known, the left hand sides of the Equations (2.38) can be computed exactly and they can be used to determine the values of the interface velocities satisfying the GCL. However, it should be noticed that, even if Equation (2.38) allows to automatically respect the GCL constraint and therefore it is the most natural way to compute the interface velocities, this choice is not unique [98, 41].

The differential relations (2.38) can be used to complete the system of the Euler equations in the ALE framework (2.21):

$$\begin{cases} \dfrac{d}{dt}\left[V_i\mathsf{u}_i\right] = \displaystyle\sum_{k\in\mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i,\mathsf{u}_k,\nu_{ik},\hat{\boldsymbol{\eta}}_{ik},\eta_{ik}) + \Phi^{\partial}(\mathsf{u}_i,\nu_i,\hat{\boldsymbol{\xi}}_i,\xi_i)\,, & i\in\mathcal{K} \\ \dfrac{dV_{ik}}{dt} = \nu_{ik} & k\in\mathcal{K}_{i,\neq} \\ \dfrac{dV_{j,\partial}}{dt} = \nu_j\,, & j\in\mathcal{K}_{\partial} \end{cases} \tag{2.39}$$

where $\mathcal{K}_{\partial}$ is the set of boundary nodes. System (2.39) consists in $5N_V + N_{ik} + N_{V,\partial}$ ODEs, with N_{ik} and $N_{V,\partial}$ the number of node-pairs and boundary nodes, respectively. As already observed, in the class of problem considered here, the positions of grid nodes are known, thus the two last equations are not coupled to the ODE system of the Euler equations. This is not the case in fluid-structure or aeroelastic problems, in which the velocities of the grid nodes depend on the

structural problem which also has to take into account the aerodynamic loads resulting from the flow field. In such a problem, the structural, fluid dynamics and mesh motion problems are fully coupled. The fluid-structure solver usually acts as a black box, which not only embeds different disciplinary softwares to solve each field independently, but also take care of exchanging the information between the different models, so that the whole problem can be solved at once [109]. However, the coupled form of System (2.39) is preferred to highlight the existence of a consistency constraint on the interface velocities. Indeed, the IVC conditions are related to the time derivative of the cell volume by the following scalar identity

$$\frac{dV_i(t)}{dt} = \sum_{k \in \mathcal{K}_{i,\neq}} \frac{dV(t)_{ik}}{dt} + \frac{dV_{j,\partial}}{dt} \,.$$

The GCL (2.37) can be discretized in time by resorting to the same BDF scheme used for the governing equation, i.e.

$$\sum_{q=-1}^{p-1} \alpha_q \frac{\Delta V_i^{n-q}}{\Delta t} = \nu_i^{n+1} + \sum_{k \in \mathcal{K}_{i,\neq}} \nu_{ik}^{n+1} \,, \tag{2.40}$$

which is usually denominated Discrete Geometric Conservation Law (DGCL) [44]. More precisely, it is the DGCL of the numerical scheme (2.32), since no DGCL exists *per se* but it is necessarily associated to the specific numerical scheme for which it has been derived [63].

As shown above, the DGCL can be split in two relations expressing the domain and the boundary contributions. Therefore, the fully discrete system of the governing equations reads

$$\begin{cases} \displaystyle\sum_{q=-1}^{p} a_q V_i^{n-q} \mathsf{u}_i^{n-q} = \Delta t \Big[\sum_{k \in \mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik})^{n+1} \\ \qquad\qquad + \ \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i)^{n+1} \Big] \,, & i \in \mathcal{K} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{ik}^{n-q} = \Delta t \ \nu_{ik}^{n+1} \,, & k \in \mathcal{K}_{i,\neq} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{j,\partial}^{n-q} = \Delta t \ \nu_j^{n+1} \,, & j \in \mathcal{K}_{\partial} \end{cases} \tag{2.41}$$

where ΔV_{ik}^n and $\Delta V_{j,\partial}^n$ are the volume swept by the interfaces $\partial \mathcal{C}_{ik}$ and $\partial \mathcal{C}_i(t) \cap \partial\Omega$ during the time step from t^n to t^{n+1}. It is worth noticing that the system (2.41) does not contain explicitly the grid velocity $\boldsymbol{v}$. Indeed, the GCL is fulfilled by computing the interface velocities ν_{ik} and ν_i according to the IVC condition [102].

2.5.1 Interface velocity computation

In System (2.41), the GCL compliant interface velocity ν_{ik} is computed from the value of the swept volume ΔV_{ik}, which is

$$\Delta V_{ik} = \int_{t^n}^{t^{n+1}} \int_{\partial \mathcal{C}_{ik}(t)} \boldsymbol{v}(t) \cdot \boldsymbol{n}_i(t) \,.$$

The computation of the swept volume is performed by splitting the interface $\partial\mathcal{C}_{ik}$ into the same triangular elemental contributions $\partial\mathcal{C}^m_{ik,f}$ introduced in Subsection 2.3.1, namely

$$\Delta V_{ik} = \sum_{m\in\varepsilon_i\cap\varepsilon_k}\sum_{f\in\mathcal{F}_{ik,m}} \Delta V^m_{ik,f}\,,$$

as shown in Figure 2.6. Assuming a constant velocity of the grid points during the time step, the volume swept by the elemental interface during the time step $\Delta t^n = t^{n+1} - t^n$ can be computed as

$$\Delta V^{m,n}_{ik,f} = \frac{\boldsymbol{v}^n_m + \boldsymbol{v}^n_f + \boldsymbol{v}^n_{ik}}{3}\int_{t^n}^{t^{n+1}} \boldsymbol{\eta}^m_{ik,f}(t) \tag{2.42}$$

where $\boldsymbol{v}^n_j = (\boldsymbol{x}^{n+1}_j - \boldsymbol{x}^n_j)/\Delta t^n$ are the velocities of the barycenters of the element $\boldsymbol{x}_m$, of the face $\boldsymbol{x}_f$ and of the edge $\boldsymbol{x}_{ik}$, for $j = m$, $j = f$ and $j = ik$, respectively.

To compute the integral in (2.42), an explicit expression of the integrated normal portion $\boldsymbol{\eta}^m_{ik,f}(t)$ for $t^n \le t \le t^{n+1}$ is required. Given the description of the point position during the time step as $\boldsymbol{x}_j(t) = \boldsymbol{x}^n_j + \boldsymbol{v}^n_j(t-t^n)$ (for $j = m, f, ik$), after some algebra it results

$$\begin{aligned}\boldsymbol{\eta}^m_{ik,f}(t) = \boldsymbol{\eta}^n_{ik} &+ \frac{t-t^n}{\Delta t^n}\left[-3\boldsymbol{\eta}^n_{ik} + 4\boldsymbol{\eta}^{n+\frac{1}{2}}_{ik} - \boldsymbol{\eta}^{n+1}_{ik}\right] \\ &+ 2\left(\frac{t-t^n}{\Delta t^n}\right)^2\left[\boldsymbol{\eta}^n_{ik} - 2\boldsymbol{\eta}^{n+\frac{1}{2}}_{ik} + \boldsymbol{\eta}^{n+1}_{ik}\right]\end{aligned} \tag{2.43}$$

where the indices f and m are omitted in the right side to simplify the notation and $\boldsymbol{\eta}^{n+\frac{1}{2}}_{ik}$ is defined as

$$\begin{aligned}\boldsymbol{\eta}^{n+\frac{1}{2}}_{ik} = 1/4(\boldsymbol{\eta}^n_{ik} + \boldsymbol{\eta}^{n+1}_{ik}) - 1/8\,\big[&(\boldsymbol{x}^n_f - \boldsymbol{x}^n_m)\times(\boldsymbol{x}^{n+1}_{ik} - \boldsymbol{x}^{n+1}_m) \\ &+ (\boldsymbol{x}^{n+1}_f - \boldsymbol{x}^{n+1}_m)\times(\boldsymbol{x}^n_{ik} - \boldsymbol{x}^n_m)\big]\,.\end{aligned}$$

Finally, integrating in time and substituting the expressions for the velocities of the points, the equation for the volume swept by the interface $\partial\mathcal{C}^m_{ik,f}$ during the time step Δt^n is obtained:

$$\begin{aligned}\Delta V^{m,n}_{ik,f} = \frac{1}{18}(\boldsymbol{x}^{n+1}_m - \boldsymbol{x}^n_m + \boldsymbol{x}^{n+1}_f - \boldsymbol{x}^n_f + \boldsymbol{x}^{n+1}_{ik} - \boldsymbol{x}^n_{ik})\cdot \\ (\boldsymbol{\eta}^{m,n}_{ik,f} + \boldsymbol{\eta}^{m,n+1}_{ik,f} + 4\boldsymbol{\eta}^{m,n+\frac{1}{2}}_{ik,f})\,.\end{aligned} \tag{2.44}$$

With a similar procedure, an expression for the volume swept by a boundary interface is obtained. First of all, the integrated boundary normal portion $\boldsymbol{\xi}^{m^\partial}_{i,f^\partial}(t)$ for $t^n \le t \le t^{n+1}$ is expressed similarly to Equation (2.43), whit

$$\begin{aligned}\boldsymbol{\xi}^{n+\frac{1}{2}}_{i,f^\partial} = 1/4(\boldsymbol{\xi}^n_i + \boldsymbol{\xi}^{n+1}_i) \pm 1/8\,\Big[&(\boldsymbol{x}^n_m - \boldsymbol{x}^n_f)\times(\boldsymbol{x}^{n+1}_i - \boldsymbol{x}^{n+1}_f) \\ &+ (\boldsymbol{x}^{n+1}_m - \boldsymbol{x}^{n+1}_f)\times(\boldsymbol{x}^n_i - \boldsymbol{x}^n_f)\Big]\end{aligned}$$

where the sign $\pm$ depends on whether i is the first $(-)$ or second $(+)$ node of the node-pair corresponding to the face f^∂, which is an edge of the boundary triangle.

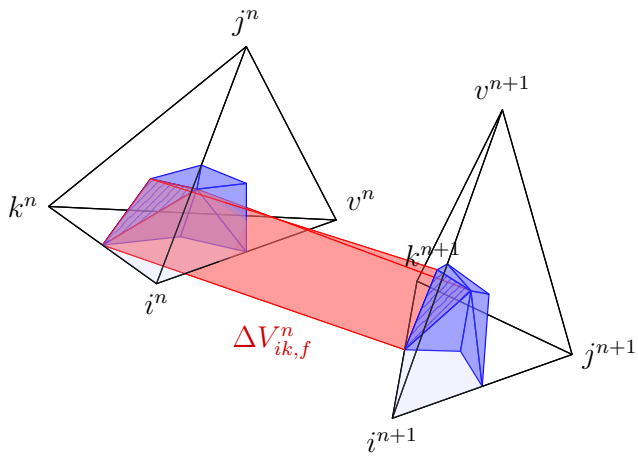

Figure 2.6: Volume swept by the elemental interface $\partial\mathcal{C}^m_{ik,f}$, during the time level $t^n \leq t \leq t^{n+1}$, because of the displacement of the nodes from their positions $\boldsymbol{x}^n$ to $\boldsymbol{x}^{n+1}$. The pattern ▨ highlights the elemental interface $\partial\mathcal{C}^m_{ik,f}$, while the red color ■ depicts the swept volume $\Delta V^{m,n}_{ik,f}$.

Finally, the equation for the volume swept by the boundary interface reads

$$\begin{aligned}\Delta V^{m^\partial,n}_{i,f^\partial} = \frac{1}{18}\big(&\boldsymbol{x}^{n+1}_{m^\partial} - \boldsymbol{x}^{n}_{m^\partial} + \boldsymbol{x}^{n+1}_{f^\partial} - \boldsymbol{x}^{n}_{f^\partial} + \boldsymbol{x}^{n+1}_{i} - \boldsymbol{x}^{n}_{i}\big)\\ &\cdot\big(\boldsymbol{\xi}^{m^\partial,n}_{i,f^\partial} + \boldsymbol{\xi}^{m^\partial,n+1}_{i,f^\partial} + 4\boldsymbol{\xi}^{m^\partial,n+\frac{1}{2}}_{i,f^\partial}\big)\,.\end{aligned} \tag{2.45}$$

Chapter 3

Conservative ALE scheme for adaptive meshes

The conservative ALE scheme derived in the previous chapter is here extended to adaptive grids with variable connectivity. To correctly take into account the topology changes due to local mesh modifications, a three-steps procedure is used to interpret local grid alterations as local continuous deformations of the finite volumes that compose the mesh, as described in Section 3.1. Particular care has to be taken in dealing with removed nodes, because the interface velocities associated to the removed interfaces are not null, even if their current surface is null. In Section 3.2, the full system of the governing equations for adaptive meshes with variable connectivity is given.

3.1 Continuum interpretation of 3D grid topology modifications

In the previous chapter, the implicit finite volume scheme for the ALE formulation of the 3D Euler equations was derived. To ensure conservativeness, the flow equations are complemented by the IVC conditions that allow to automatically satisfy the DGCL on moving domain. More precisely, the change in position and shape of the finite volumes are easily taken into account by computing the interface velocities as the sum of the volume swept by the interfaces during the time step. Therefore, if some IVC-compliant interface velocities can be computed also when topology changes occur, the proposed conservative ALE scheme can be extended straightforwardly to local mesh adaptation techniques, like node insertion, node deletion and edge swap. A possible choice for computing the interface velocities if local grid modifications occur is thoroughly explained in the follows.

To guarantee the conservativeness and the monotonicity of the scheme, the volume and shape modifications due to mesh modifications have to correctly be

taken into account when projecting the solution on the new grid. Alternatively, the conservation of mass, momentum and energy has to be explicitly imposed and the total variations of the solution might need to be limited. For fixed topology grid, this problem is successfully tackled and different examples of conservative ALE scheme for moving boundaries can be found in the literature, as for instance [104, 95, 68, 103, 94]. When grid adaptation techniques are applied and the grid connectivity changes, in principle, the solution obtained at the previous time step has to be interpolated onto the new grid, but unfortunately this procedure undermines the conservativeness and the monotonicity of the scheme if spatial accuracy is preserved. Moreover, additional difficulties might arise when multi-step high-order integration schemes are used. For a p-steps time integrator, the interpolation procedure has to be repeated p times because the integration at the current time level requires the values of the solutions at the previous p time levels [128].

A different strategy is pursued in the present thesis to avoid the interpolation of the solution between different grids. The underlying idea is to give an interpretation of the grid connectivity changes that occur due to mesh adaptation as series of fictitious continuous deformations of the finite volumes. This strategy was proposed and applied to two-dimensional grids by Guardone and collaborators [59, 71, 70] and it is extended here to three-dimensional tetrahedral grids. In the proposed approach the volume swept by the interface velocities is split into two contributions: the deformation, denoted as ΔD, and the adaptation one, denoted as ΔA. The former contribution is the standard swept volume due to mesh movement and deformation and it is computed as described in the Subsection 2.5.1. The second contribution is due to local grid connectivity changes and it is computed by a three-steps sequence of continuous deformations as described in the next subsection. Accordingly, the total volume swept by the interface $\partial\mathcal{C}_{ik}$ during the time step n is $\Delta V_{ik}^n = \Delta D_{ik}^n + \Delta A_{ik}^n$. These additional deformations due to grid adaptation result in additional fictitious ALE fluxes, which can be easily taken into account by standard ALE techniques, without requiring any explicit interpolation of the solution over the new grid. Admittedly, as observed in [71], the ALE mapping is indeed equivalent to an interpolation, but it does not require any special treatment to guarantee the conservativeness, the monotonicity and accuracy of the scheme.

As shown in the next subsection, the three-steps procedure proposed here allows to compute the interface velocities in a IVC-compliant fashion, therefore the conservation of flow variables is inherently guaranteed by scheme construction. Moreover, the implementation of high-order multi-step time schemes poses no problems because the solutions at previous time levels can be retrieved easily.

3.1.1 Three steps procedure

To take into account the volume changes due to local mesh adaptation, a three-steps procedure is introduced. The fundamental idea is to describe the grid modification as a sequence of fictitious continuous deformations for which the volumes swept by cell interfaces can be computed by Equations (2.44) and (2.45). This can be accomplished by the following three-steps procedure, in which a fictitious time ζ is used to better distinguish the different phases, with the initial configuration at $\zeta=0$ and the final one at $\zeta=1$.

1. *Collapse*, $0<\zeta<1/2$: the elements involved in the local modification collapse over an arbitrary point.

2. *Connectivity change*, $\zeta=1/2$: when all involved elements reach null volumes, the node is inserted or deleted. This change does not generate any numerical fluxes, because it takes place at null volumes and interfaces.

3. *Expansion*, $1/2 < \zeta < 1$: the elements expand to their final configuration (if they were not deleted at $\zeta = 1/2$).

To better understand the three-steps procedure, a two-dimensional example is presented before a detailed description of its application to three-dimensional grid modifications. In 2D, the proposed procedure can be sketched more clearly and the fundamental idea can be better illustrated.

2D example

The three-steps procedure is now introduced by a two-dimensional example. Figure 3.1 shows its application to the edge split, which consists in splitting the original edge into two edges by the insertion of a new node. Suppose that the edge e_{ik} should be split by inserting the new node j. The edge e_{ik} is shared by two elements (the triangle i-k-v_1 and i-k-v_1), which have to be split to restore the correct topology for the triangular grid. Figure 3.1 displays also the finite volumes associated to the node i and k and how it changes during the three-steps procedure. In 2D, the interface between two finite volumes consists of line segments which during deformation sweep a certain area (a volume in 3D). The first step consists in collapsing the elements sharing the edge, which compose the quadrilateral i-k-v_1-v_2. The collapse point is arbitrarily chosen as the position of the new node j. Then, when the elements have a null area, the new node is inserted and the topology is modified. Finally, the nodes i, k, v_1, v_2 are moved back to their original positions and the new finite volume associated to j expands to reach the final configuration. The finite volume associated to the nodes i, k, v_1 and v_2 have changed during the process, but, as clearly shown in Figure 3.2, the modifications are limited to the portion of the interfaces inside the quadrilateral i-k-v_1-v_2.

Detailed description in 3D

The three-steps procedure is now described more thoroughly by referring to Figure 3.3, which shows its application to an edge split in 3D. This modification involves all the elements that share initially the edge to split. With reference to Figure 3.3, consider that the edge between node i and k has to be split to insert the new node j. Hence, according to the dual mesh definition, the four elements shown in the picture and the finite volumes associate to nodes i, k, v_1, v_2, v_3, v_4 have to be modified to generate the new volume associated with the new node.

Once the involved elements have been identified, the first phase can be carried out. The collapse from the original configuration to an arbitrary point is simply a continuous deformation and therefore the swept volumes can be computes with Equation (2.44) and (2.45).

At this point, the new node can be inserted and connected with the existing ones to generate new tetrahedra. Since the connectivity change occurs when the

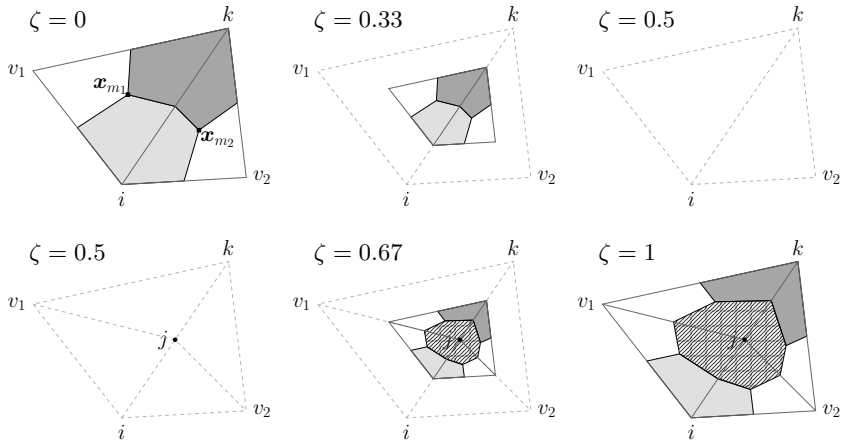

Figure 3.1: Three-steps procedure applied to the split of edge e_{ik} in 2D. The dashed grey lines show the grid connectivity in the original/final configuration, i.e. the non-collapsed configuration, for $0 < \zeta < 1$, while the portions of the finite volumes associated to i, k and j are shown with light grey □, dark grey ■ and the pattern ▨ , respectively. In the first row the collapse phase is depicted: the quadrilateral i-k-v_1-v_2, composed by the elements that share the edge at $\zeta = 0$, is collapsed over the mid-point of the edge e_{ik}. When it reaches a null area, the connectivity is changed ($\zeta = 0.5$): the new point j is inserted, the edge e_{ik} is split into two edges (i-j and k-j) and two new edges are created to connect j to v_1 and v_2. The second row displays the expansion procedure: the nodes i, k, v_1, v_2 return to their original positions to reach the final configuration (at $\zeta = 1$).

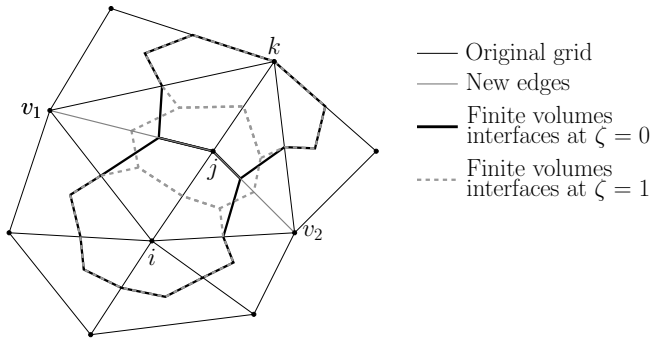

Figure 3.2: Local character of the edge slit. The initial and the final configuration for the split of the edge e_{ik} described in Figure 3.1 are shown together, along with some elements external to quadrilateral i-k-v_1-v_2. The boundaries of the finite volumes associated to i and k are drawn with different lines to display the modifications due to the edge split. No modifications occur outside the quadrilateral i-k-v_1-v_2.

elements have null volume, and the interfaces have null area, no volume is swept by any interface. Consequently, this change has no effects in terms of interface velocities.

Finally, the elements expand towards the final configuration with a continuous deformation, which allows to easily compute the swept volume with Equation (2.44) and (2.45). As shown in the last two pictures of Figure 3.3, differently from the collapse phase, a new finite volume is created and also the volume swept by its interfaces has to be considered.

The complete contribution to the swept volume due to adaptation ΔA is given by the sum of the volume swept during the collapse and the expansion phases. It is important to notice the local character of the modification. Similarly to what happens in 2D, see Figure 3.2, the external faces of the polyhedron formed by the union of the involved elements remain unchanged. Indeed, since no variation occurs in the grid elements outside the polyhedron, no variation occurs in the finite volumes either. Then, the volume swept by the interfaces located outside the polyhedron during the whole procedure is null, because the interfaces swept the same volume but with opposite signs during the collapse and the expansion steps. For instance, for the split of the edge i-k,

$$\Delta A_{lq} = \Delta A_{lq}^{\mathsf{col}} + \Delta A_{lq}^{\mathsf{exp}} = \Delta A_{lq}^{\mathsf{col}} - \Delta A_{lq}^{\mathsf{col}} = 0 \quad \forall\, \partial\mathcal{C}_{lq} \notin \mathsf{poly}(\mathcal{B}_{ik}^{m})$$

where $\mathsf{poly}(\mathcal{B}_{ik}^{m})$ denotes the polyhedron composed by the element bubble of the edge i-k, i.e. all elements having edge e_{ik} in common, and the subscripts col and exp indicate the collapse and expansion phase, respectively. For this reason, the elements outside the polyhedron are kept unchanged and they are not collapsed nor expanded during the three-steps procedure.

This three-steps procedure is general and it can be applied also to other local adaptation techniques. Obviously, the elements to be collapsed and expanded are different depending on the grid modification, as described in Table 3.1 for all the local adaptation techniques used in the present work. Figure 3.4 shows how this procedure is applied to the edge collapse, during which the node j is collapsed over the node i. Differently from the edge split procedure, in this case the initial configuration contains elements that have to be removed to comply with the new topology. Indeed, they are deleted at $\zeta = 1/2$ and they do not take part in the expansion phase. Also the finite volume connected to the node j is deleted at $\zeta = 1/2$. Nevertheless, it should be noticed that during the collapse phase the interfaces of $\mathcal{C}_j$ sweep a non-null volume, which generates non-null interface velocities. This fact has to be carefully taken into account, as described in Subsections 3.1.3 and 3.2.2.

A final remark has to be done regarding the collapse point. As observed before, it can be arbitrarily chosen and the DGCL fulfillment does not depend on this choice. However, the larger is the volume swept by the interfaces during the collapse and expansion to and from this position, the greater the value of the interface velocities is. An inappropriate choice of the collapse point, therefore, can introduce large perturbations in the governing equations. Thus, in this work the collapse point is chosen in the proximity of the middle of the polyhedron of elements involved in the grid modification, but no exact minimization of the swept volume is performed. The collapse points chosen for each used adaptation technique are listed in Table 3.1.

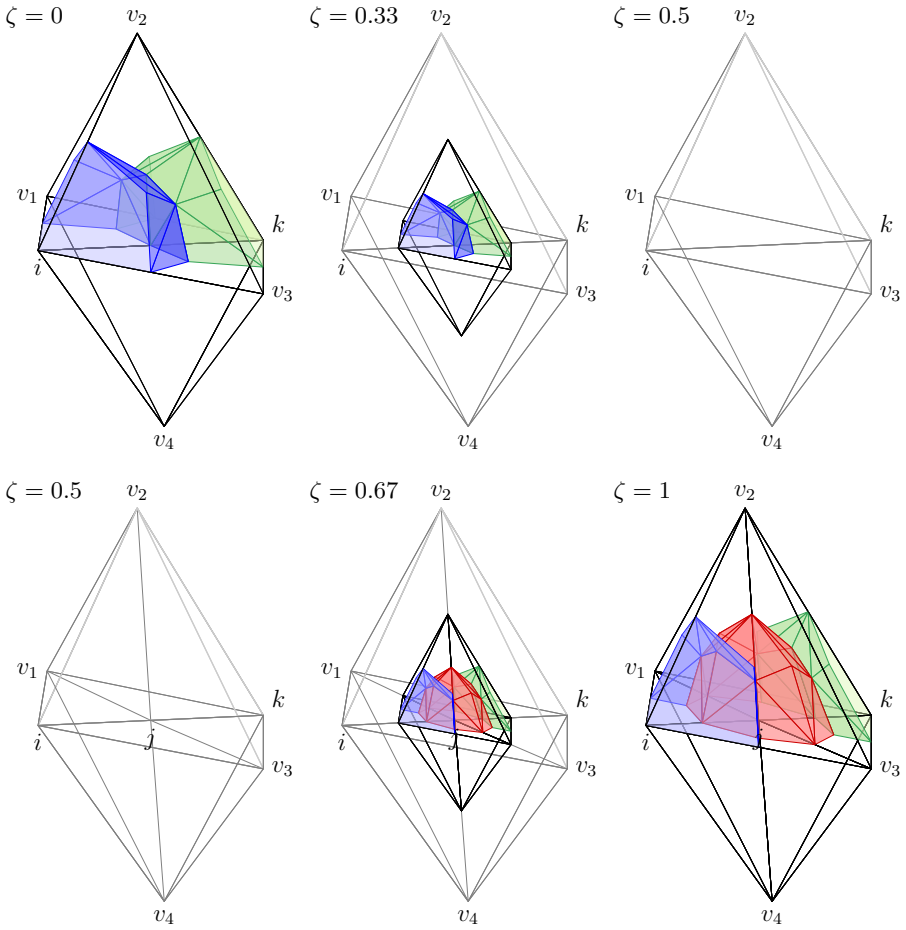

Figure 3.3: Three-steps procedure applied to the split of the edge e_{ik}. The grey lines show the grid connectivity in the original/final configuration, i.e. the non-collapsed configuration, for $0 < \zeta < 1$. The portions of the finite volumes pertaining to the polyhedron i-k-v_1-v_2-v_3-v_4 and associated to the nodes i, k and j are shown, respectively, with blue ■/■, green ■/■, and red ■/■ (the darker colors refer to the boundaries of the finite volumes). The first row depicts the collapse phase: the polyhedron is collapsed over the mid-point of the edge e_{ik}. The connectivity change occurs at $\zeta = 0$, when the surface of the involved interfaces has null area. The second row displays the expansion procedure: the nodes i, k, v_1, v_2, v_3, v_4 return to their original positions and the new finite volume $\mathcal{C}_j$ expands to reach the final configuration (at $\zeta = 1$).

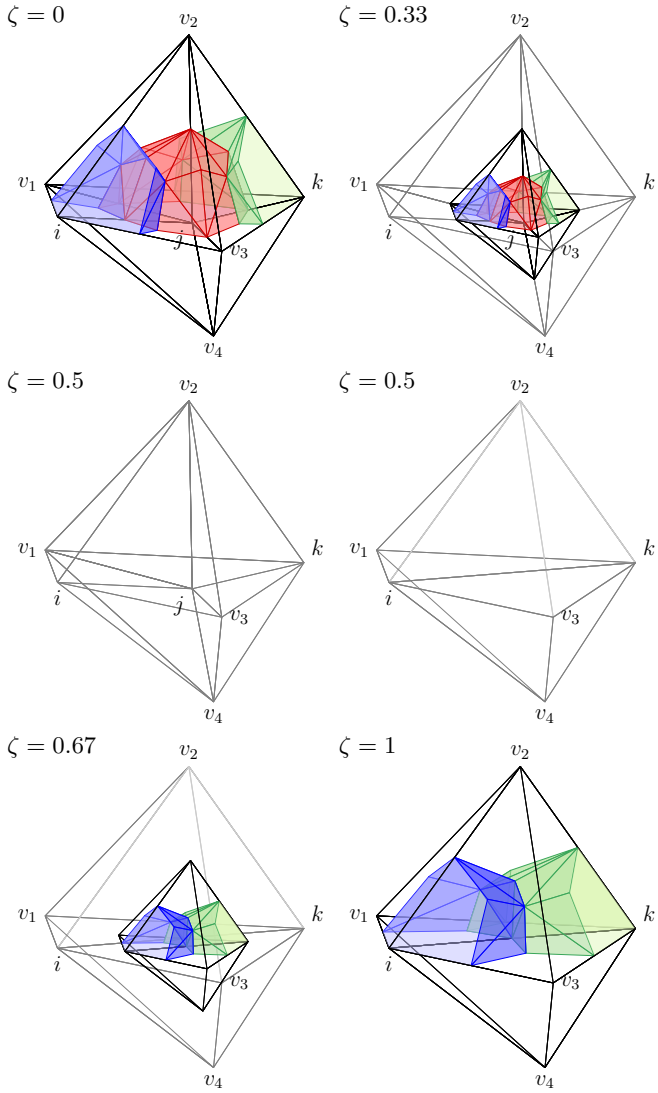

Figure 3.4: Three-steps procedure applied to the collapse of the edge e_{ij}. The grey lines show the grid connectivity in the original/final configuration, for $0 < \zeta < 1$. The portions of the finite volumes pertaining to the polyhedron i-k-v_1-v_2-v_3-v_4 and associated to the nodes i, k and j are shown, respectively, with blue ■/■, green ■/■, and red ■/■ (the darker colors refer to the boundaries of the finite volumes). The first row depicts the collapse of the involved elements over the point j, which reach a null volume. At $\zeta = 0.5$ (second row), the node j is collapsed over the node i. The elements that share the edge e_{ij} are deleted, while for the other elements the node j is substituted by the node i. In the expansion phase (third row), only the elements that are not deleted are expanded to reach the final configuration (at $\zeta = 1$).

Adaptation	Polyhedron	Collapse point
Edge split	Elements sharing the edge	New node
Element split	Element to split	New node
Delaunay insertion	Elements in the cavity	New node
Edge collapse	Elements sharing the node to be deleted	Node to be deleted

Table 3.1: Three-steps procedure details for the considered local mesh adaptation techniques. For each technique adopted in the present work, the elements composing the polyhedron that delimits the finite volumes influenced by the grid modification are indicated, along with the chosen collapse points.

3.1.2 Edge creation

During the three-steps procedure, the connectivity change may require the creation of new grid edges to restore the correct topology. Consequently, new node-pairs, and therefore new interfaces, are generated to comply with the dual mesh definition. These modifications occur at $\zeta = 1/2$, so the new interfaces sweep a non-null volume only during the expansion phase. Despite this, no difficulties arise in applying the three-steps procedure and the numerical fluxes and the interface velocities can be computed normally as shown in Subsection 2.3.2 and Section 2.5, respectively.

3.1.3 Edge deletion

When the connectivity change requires to delete an edge i-k, the corresponding portion of the interface $\partial\mathcal{C}_{ik}$ is collapsed to reach a null area at $\zeta = 1/2$ and then it is no more expanded. Suppose that this modification occurs in the adaptation phase performed between the time level t^n and t^{n+1}, then the interface has a null area for any $t^m \geq t^{n+1}$ and hence $\eta_{ik}^m = 0$. However, during the collapse phase the interface possibly sweeps a non-null volume $\Delta A_{ik}^{\mathsf{col}} \neq 0$, so the interface velocity ν_{ik}^m is different from zero. Indeed, if no other grid modifications nor deformations occur, according to Equations (2.41) the interface velocity is $\nu_{ik}^{n+1} = \alpha_{-1}\Delta A_{ik}^{n+1}/\Delta t$. Moreover, for a p-steps integration scheme the interface velocity ν_{ik}^m depends on the volume swept during the previous p time levels, then $\nu_{ik}^m \neq 0$ for $m \leq n + p + 1$.

For what concerns the integrated velocity for the removed interface, a distinction between the magnitude and the unit vector is made. Because of the null area of integration, $\eta_{ik}^m = 0$ for $t^m \geq t^{n+1}$, while the normal unit vector can be defined as $\hat{\boldsymbol{\eta}}_{ik}^m = \lim_{t \to t^{n+1}} \hat{\boldsymbol{\eta}}_{ik}(t)$ and it is taken here as $\hat{\boldsymbol{\eta}}_{ik}^{n+1} = \hat{\boldsymbol{\eta}}_{ik}^n$. Hence, the numerical flux associated to a deleted node-pair reads

$$\Phi_{ik}^{n+1} = \Phi(\mathsf{u}_i^{n+1}, \mathsf{u}_k^{n+1}, \nu_{ik}^{n+1}, \hat{\boldsymbol{\eta}}_{ik}^{n+1}, 0)\,.$$

The null value of η_{ik} affects the expressions of the integrated numerical fluxes given in Subsection 2.3.2 as explained below. The first term in Equation (2.28)—as well as the first one in Equation (2.27)—is null, thus the first order flux becomes

$$\Phi_{ik}^{\mathsf{I}}(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1} = -\frac{\mathsf{u}_i^{n+1} + \mathsf{u}_k^{n+1}}{2}\nu_{ik}^{n+1} - \frac{1}{2}|\widetilde{\mathsf{A}}|(\mathsf{u}_k - \mathsf{u}_i)\,,$$

where the Roe matrix is composed only by the ALE contribution and it does not depend on $\widetilde{\mathsf{u}}$ and $\hat{\boldsymbol{\eta}}_{ik}$. The Eulerian part of the Jacobian matrix $\mathsf{A}^{\mathrm{EUL}}$ is

proportional to $\boldsymbol{\eta}_{ik} = \eta_{ik}\hat{\boldsymbol{\eta}}_{ik}$ and it is indeed null, so $\mathsf{A}^{\mathrm{ALE}} = -(\boldsymbol{v}\cdot\boldsymbol{n})\mathbb{I}^5$. Therefore the eigenvectors matrices are then identity matrices and the Roe matrix reduces to

$$\widetilde{\mathsf{A}}(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}_{ik}, 0, \nu_{ik})^{n+1} = -\nu_{ik}^{n+1}\mathbb{I}^5 .$$

As a result, the high-resolution fluxes of Equation (2.30) can be evaluated for a deleted node-pair as

$$\Phi_{ik}^{\mathsf{HR}} = -\frac{\mathsf{u}_i^{n+1} + \mathsf{u}_k^{n+1}}{2}\nu_{ik}^{n+1} - \frac{|\nu_{ik}^{n+1}|}{2}\left[\Upsilon - \mathbb{I}^5\right](\mathsf{u}_k - \mathsf{u}_i) \tag{3.1}$$

without requiring any special additional routine.

However, a modification of the entropy fix is required, because the Mach number of the intermediate state is singular for $\hat{\boldsymbol{\eta}}_{ik} = 0$. Therefore the threshold $\widetilde{\delta}$ in Equation (2.29) is modified as follows

$$\widetilde{\delta} = \begin{cases} 0.2\,\mathrm{M}(\widetilde{\mathsf{u}}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik}, \nu_{ik}) & \text{if } \eta_{ik} > 0, \\ 0.2\,|\nu_{ik}| & \text{if } \eta_{ik} = 0. \end{cases}$$

To sum up, the numerical flux associated to a deleted node-pair consists only of the ALE contribution. The balance of the conservative variables is enforced by means of the interface velocities which are different from zero even if the interface has been collapsed. The number of time steps required by the interface velocity associated to a removed node-pair to become identically null depends on the adopted time integration scheme. Furthermore, the DGCL has an intrinsically differential nature that, as shown in [70], makes the interface velocities influenced by the history of the volume deformation independently of the time integrator.

3.1.4 Special care in node-pair modification

An important remark has to be done about the distinction between grid edge and node-pair when grid topology changes. As defined in Section 2.3, a node-pair is defined as a couple of interacting nodes. Although for a finite volume dual-mesh discretization it is natural to associate the node-pair i-k to the grid edge i-k, particular care has to be taken when re-constructing the topology after local grid adaptation.

Consider for instance the sequence of grid modifications shown in Figure 3.5 that affects the same group of elements during the same time step Δt^n. First, the edge e_1 between the node i-k is split, then the new nodes i,k and j are moved and finally the node j is removed. In the last modification, a new edge e_2 is created to connect again nodes i and k. From the point of view of the grid topology, it is not necessary to recognize that an edge connected the two nodes has already existed and it was named e_1. In fact, the adaptation algorithm usually generates the new edge e_2 and it does not look for any relations with the previous edge, as described in next chapter.

On the contrary, according to node-pair definition, the nodes i and k form one and only one node-pair. Thus, the algorithm is required to identify this situation to restore the previous node-pair, along with the associated interface. Only one interface velocity has to be computed, by summing the volume swept by the interface due to the different modifications.

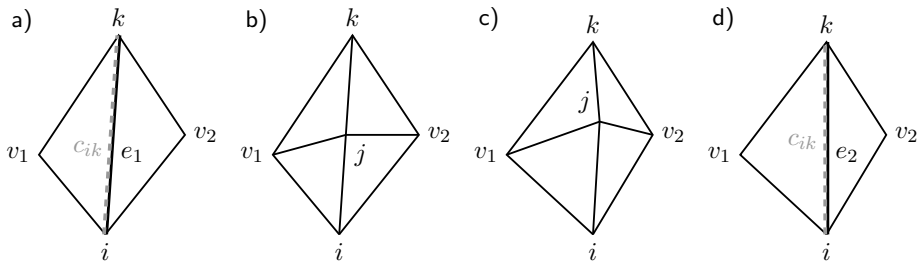

Figure 3.5: Sequence of modifications involving the same node-pair during the time step Δt^n. Picture a shows the initial configuration: the edge e_1 connects the nodes i and k, which form the node-pair c_{ik}. Then the edge is split (Picture b), the nodes i, j and k are moved (Picture c) and, finally, the node j is collapsed over k (Picture d). In this last operation, a new edge e_2 is created to connected the nodes i and k, which have been already connected by the node-pair c_{ik} at the beginning of the time step. Thus, the node-pair c_{ik} has to be restored, along with its interfaces and swept volumes.

The same care has to be taken for grid modifications in successive time steps. As already explained, if a p-steps time integrator is used, the removed interfaces enter the governing equations for p time levels after they have been removed. Hence, when creating a new edge between two existing nodes, a search among the interfaces removed in the previous p time steps has to be carried out to check if a node-pair connecting the same two nodes has already existed and, thus, has to be restored.

3.2 ALE scheme with variable topology

The finite volume node-pair formulation of the governing equations within the ALE framework (2.41) is now modified to deal with variable grid topology. To this end, the notation is improved to take into consideration that the grid and the connectivity between finite volumes and grid entities change over time. For instance, the set of nodes that compose the grid at time level t^{n+1} is denoted as $\mathcal{K}^{n+1}$.

Moreover, a subscript is introduced to state the time interval at which the variable is considered, by using the standard notation according to which square and round brackets indicate respectively to include or exclude the endpoint of the interval. The set of finite volumes that shared a portion of their boundary with $\mathcal{C}_i$ in the interval from t^{n+1-p} to t^{n+1} but that do not share anymore any portions at time level t^{n+1} is thus denoted as

$$\mathcal{K}_{i,\neq}^{[n-p,n+1)} = \{k \in \mathcal{K}, k \notin \mathcal{K}_{i,\neq}^{n+1} \quad \text{such that} \quad \nu_{ik}^{n+1} \neq 0\}\,.$$

As observed in the previous section, this set is non-null when some edges connected to i have been removed. It is important to notice that this set includes also the interfaces that, because of successive modifications, were created and deleted during the same time interval, i.e. that were not present at t^{n+1-p} and are not present at t^{n+1} but have been existed in between. Similarly, $\mathcal{K}_{i,\neq}^{[n-p,n+1]}$ is the whole set of the finite volumes that share or have shared their boundary with $\mathcal{C}_i$, i.e. $\mathcal{K}_{i,\neq}^{[n-p,n+1]} = \mathcal{K}_{i,\neq}^{[n-p,n+1)} \cup \mathcal{K}_{i,\neq}^{n+1}$.

To fulfill the DGCL, the volume swept by an interface during the whole time interval can be computed by summing the contributions of all the collapse and expansion phases experienced by the interface. Although this choice is not mandatory, it can be implemented in an extremely straightforward way. Indeed, after each local modification, the volume swept by all the involved interfaces is computed and summed to the one at the current time level. At the end of the time level, the whole swept volume is stored, so it is possible to retrieve it during the computation of the interface velocities at next time levels.

In the following subsections, the modifications to the system of governing equations (2.41) required by node insertion and deletion are described, both in case of domain or boundary adaptation. Finally, the complete system of governing equations for adaptive grids is given.

3.2.1 ALE scheme for a new node

When a new node is added to the triangulation, a new equation to enforce the balance of the conservative variables over the new finite volume has to be added to the governing equation system. Similarly, additional IVC conditions are required to compute the new interface velocities. As stated in Subsection 3.1.2, the computation of the numerical fluxes and the interface velocities requires no modifications with respect to the equations presented in the previous chapter. If a new node i is inserted between the time step t^n and t^{n+1}, the volume of the new finite volume and its interfaces are simply null for all time steps previous than t^{n+1}. Therefore, the additional governing equations due to the domain node insertion read

$$\begin{cases} a_{-1} V_i^{n+1} \mathsf{u}_i^{n+1} = -\Delta t \displaystyle\sum_{k \in \mathcal{K}_{i,\neq}^{n+1}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik})^{n+1} & i \text{ new} \\ \alpha_{-1} \Delta V_{ik}^{n+1} = \Delta t \; \nu_{ik}^{n+1} , & k \in \mathcal{K}_{i,\neq}^{n+1} . \end{cases} \tag{3.2}$$

The previous equations allow to compute the solution at the new node as a function of the actual metrics, without requiring the knowledge of any previous solutions.

In the case of a boundary node insertion, the contributions of the boundary fluxes and the interface velocities have to be taken into account in addition to the domain one. If the node i was inserted on the boundary, the new finite volume has a portion of its boundary lying on the domain boundary $\partial\Omega$ and shares the rest of its boundary with other finite volumes. Thus, the governing equations for the new node inserted over the boundary are

$$\begin{cases} a_{-1} V_i^{n+1} \mathsf{u}_i^{n+1} = -\Delta t \Big[\displaystyle\sum_{k \in \mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik})^{n+1} \\ \qquad\qquad + \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i)^{n+1} \Big] , & i \in \mathcal{K}_{\partial}^{n+1} \text{ new} \\ \alpha_{-1} \Delta V_{ik}^{n+1} = \Delta t \; \nu_{ik}^{n+1} , & k \in \mathcal{K}_{i,\neq}^{n+1} \\ \alpha_{-1} \Delta V_{i,\partial}^{n+1} = \Delta t \; \nu_i^{n+1} , & i \in \mathcal{K}_{\partial}^{n+1} \text{ new} \end{cases} \tag{3.3}$$

where $\mathcal{K}_{\partial}$ highlights that i is a boundary node.

The systems of Equations (3.2) and (3.3) allow to treat in a conservative way the node insertion, by computing the volume changes due to the grid modification

through the three-steps procedure described in Section 3.1, which ensures the fulfillment of the DGCL.

3.2.2 ALE scheme for a removed node

When a node is removed between the time level t^n and t^{n+1}, all the interfaces associated to the edges connected to it are collapsed and removed, as well as the associated control volume. However, as explained in Subsection 3.1.3, since these interfaces swept a non-null volume before collapsing, the interface velocities associated to the removed edges are not null. As a consequence, a non-null integrated numerical flux at the time level t^{n+1}, more precisely in the purely ALE contribution given by Equation (3.1). Therefore, the balance of the conservative variables has to be enforced also over the removed finite volume and the additional governing equations for a removed domain node i are

$$\begin{cases} \displaystyle\sum_{q=0}^{p} a_q V_i^{n-q} \mathsf{u}_i^{n-q} = -\Delta t \sum_{k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1}, & i \text{ removed} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{ik}^{n-q} = \Delta t \; \nu_{ik}^{n+1}, & k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)} \end{cases} \tag{3.4}$$

where the summation in the left hand side of the first equation starts from $q = 0$ since $V_i^{n+1} = 0$. The right hand side of the first equation depends on the solution on the removed node evaluated at t^{n+1}, which is required for the computation of the solution on other nodes since an implicit time scheme is used to integrate in time the governing equations. For instance, if the removed node i was connected to k by an edge that is no longer present in the grid, the flux Φ_{ik}^{n+1} is required to compute the actual solution at node k, but the computation of this flux requires the knowledge of the value of u_i^{n+1}, which is computed via Equation (3.4).

The IVC conditions, i.e. the equations on the second rows of System (3.4), are defined for the set of the nodes that have been connected to the removed node i in the previous p steps, namely for the removed interfaces. The interface velocity ν_{ik} remains different from zero for the next p time steps, i.e. $\nu_{ik}^{n+p} = \alpha_{p-1} \Delta V^{n+1} \neq 0$. Then, the additional equations for a removed node have to be enforced for the p steps successive the deletion. Since the last non-null swept volume is ΔV_i^{n+1}, at $t = n + p + 1$ the swept volumes included in the IVC conditions are all null, so the equations for the removed node i can be dropped from the system of governing equations.

Similar to the node insertion procedure, if the node to be removed i lies on boundary $\partial\Omega$ also the contribution of the boundary interfaces has to be included in the additional equations. Moreover, as observed for the domain removed interfaces in Subsection 3.1.3, also the non-null contributions associated to removed interfaces have to be considere. Indeed, although the boundary interface $\partial\mathcal{C}_i \cap \partial\Omega$ has a null area, the numerical flux across it and the associated interface velocities are not null. Similarly to $\boldsymbol{\eta}_{ik}$, the boundary integrated normal $\boldsymbol{\xi}_i$ has a null magnitude, i.e. $\xi_i^{n+1} = 0$, while its unit vector is taken as $\hat{\boldsymbol{\xi}}_i^{n+1} = \hat{\boldsymbol{\xi}}_i^{n}$. Therefore,

the additional governing equations for a removed boundary node are

$$\begin{cases} \displaystyle\sum_{q=0}^{p} a_q V_i^{n-q} \mathsf{u}_i^{n-q} = -\Delta t \sum_{k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1} \\ \qquad\qquad - \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, 0)^{n+1} \Big], \qquad i \in \mathcal{K}_{\partial}^{n+1} \text{ removed} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{ik}^{n-q} = \Delta t \; \nu_{ik}^{n+1}, \qquad k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{i,\partial}^{n-q} = \Delta t \; \nu_{i}^{n+1}, \qquad i \in \mathcal{K}_{\partial}^{[n-p,n+1)} \end{cases} \tag{3.5}$$

where the boundary flux Φ^{∂} consists only in the ALE contribution, namely Equations (2.16) for a removed boundary interface reduces to $\Phi^{\partial} = -\mathsf{u}_i \nu_i$. Since the Eulerian contribution of the fluxes is null, the boundary conditions are not imposed on the removed node, so the boundary state is taken as $\mathsf{u}_i^{\partial} = \mathsf{u}_i$.

3.2.3 Conservative ALE governing equations for adaptive grids

All modifications described in the previous subsections are now combined to obtain the conservative system of the ALE governing equations for adaptive grids, namely

$$\begin{cases} \displaystyle\sum_{q=-1}^{p} \frac{a_q V_i^{n-q} \mathsf{u}_i^{n-q}}{\Delta t} = - \sum_{k \in \mathcal{K}_{i,\neq}^{n+1}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik})^{n+1} \\ \qquad\qquad - \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i)^{n+1} \\ \qquad\qquad - \displaystyle\sum_{k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1}, \quad i \in \mathcal{K}^{n+1} \\ \displaystyle\sum_{q=0}^{p} \frac{a_q V_i^{n-q} \mathsf{u}_j^{n-q}}{\Delta t} = - \sum_{k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1} \\ \qquad\qquad - \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, 0)^{n+1}, \qquad i \in \mathcal{K}^{[n-p,n+1)} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{ik}^{n-q} = \Delta t \; \nu_{ik}^{n+1}, \qquad k \in \mathcal{K}_{i,\neq}^{[n-p,n+1]} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{i,\partial}^{n-q} = \Delta t \; \nu_{i}^{n+1}, \qquad i \in \mathcal{K}_{\partial}^{[n-p,n+1]} \end{cases} \tag{3.6}$$

where the first and the second equations expresses the balance of conservative variables for, respectively, all the nodes of the triangulation at time level t^{n+1} and for the finite volumes associated to the nodes removed between t^{n-p} and t^{n+1}, for which the interface velocities are non-null, i.e.

$$\mathcal{K}^{[n-p,n+1)} = \left\{ j \notin \mathcal{K}^{n+1} : \exists k \text{ such that } \nu_{ik}^{n+1} \neq 0 \right\}.$$

These two equations are therefore different versions of the same Euler equations and they are mutually exclusive: if the node i belongs to the actual triangulation $\mathcal{K}^{n+1}$ the first one has to be enforced, if it is removed during the previous p steps, the second one has to be solved.

The last rows of the previous system express the IVC conditions that allow to compute the cell volume changes due to mesh deformation and adaptation so that the DGCL is matched. Thanks to the described three-steps procedure, the solution can be conservatively computed within the ALE framework even if topology changes occur due to grid adaptation.

Chapter 4

Three-dimensional mesh adaptation strategy

The mesh adaptation strategy exploited to modify the grid spacing and to deal with large boundary displacements is here described. The first section explains how the target grid spacing is computed by means of an indicator based on solution gradients. This information is given as input to the external library Mmg3d which modifies the grid through a suitable mix of the local mesh adaption techniques described in Section 4.2. Then, Section 4.3 presents the algorithm to re-locate the internal grid points when the boundaries experience rigid movements and deformation. Finally, an overview of the complete numerical simulation procedure for moving-bodies problems is given in Section 4.4.

4.1 Metric construction

Mesh adaptation has proved to be a valuable tool in CFD simulations [10, 22, 46, 64, 99]. However, its effectiveness is strictly related to the capability of specifying a suitable grid spacing which allows to increase the solution accuracy without increasing excessively the number of nodes. As observed in Subsection 1.3.1, different criteria can be used to identify the regions of the grid where refinement is required and the ones where coarsening operations would not undermine the solution accuracy.

In this work, criteria based on the solution variations are chosen for their simplicity and error estimators are built using the gradient and the Hessian of relevant flow variables. The underlying idea of this kind of adaptation criteria is that the largest errors occur where the solution exhibits the largest gradients, therefore the solution accuracy can be increased by gathering mesh points in these regions.

This section explains the construction of the metric that expresses the target grid spacing. First of all, an error estimator is computed as detailed in

Subsection 4.1.1 and the target grid spacing is specified following the procedure explained in Subsection 4.1.2. Then, the grid spacing is translated into a metric field. Finally, a possible extension to adaptation criteria based on error estimates is briefly discussed in Subsection 4.1.4.

4.1.1 Error estimator computation

In several applications, error estimators based on first or second order derivatives of a relevant flow variable are well suited to drive mesh adaptation [92, 10, 23]. To efficiently relate the grid spacing to the behavior of the solution, it is necessary to compute an initial solution. For steady problems, mesh adaptation is an iterative process, in which the solution computed over the grid obtained in the previous iteration is used to compute the new adapted grid. For unsteady problems, at each time level, the initial solution is the one obtained at the previous time level or during the prediction phase, as explained in Section 4.4.

In this work, the error indicator can be based on density, pressure, temperature, Mach number or entropy, which can be easily obtained directly from the conservative variables or by means of thermodynamic relations [20]. However, in some problems of aeronautical interest it is desirable to refine the grid mesh near vortical structures and the vorticity of the solution, i.e. $\boldsymbol{\omega} = \boldsymbol{\nabla} \times \boldsymbol{u}$ where $\boldsymbol{u} = \boldsymbol{m}/\rho$ is the fluid velocity, is a more recommended choice for the error indicator. Following [49], within the node-pair finite-volume formulation presented in Section 2.3, the vorticity at the node i can be computed as

$$\boldsymbol{\omega}_i = \frac{1}{V_i} \left[\boldsymbol{\xi}_i \times \frac{\boldsymbol{m}_i}{\rho_i} + \frac{1}{2} \sum_{k \in \mathcal{K}_{i,\neq}} \boldsymbol{\eta}_{ik} \times \left(\frac{\boldsymbol{m}_i}{\rho_i} + \frac{\boldsymbol{m}_k}{\rho_k} \right) \right] . \tag{4.1}$$

After the variable chosen as sensor has been evaluated in all grid points, the gradient or the Hessian matrix are computed. Similarly to Equation (4.1), the gradient of a scalar quantity s at the node i is computed by resorting to the node-pair finite volume discretization [49] as

$$\boldsymbol{\nabla} s_i = \frac{1}{V_i} \left[s_i \boldsymbol{\xi}_i + \sum_{k \in \mathcal{K}_{i,\neq}} \frac{s_i + s_k}{2} \cdot \boldsymbol{\eta}_{ik} \right] . \tag{4.2}$$

The previous relation allows to calculate the approximated value of the gradient at each node by looping on all node-pairs sharing the node i. Since a scalar estimator error ϵ is required to compute the isotropic target metric field, at each grid node i it is taken as the norm of the gradient, i.e. $\epsilon_i = V_i \| \boldsymbol{\nabla} s_i \|$, where the volume V_i helps to avoid over-refinement.

The second derivative of the flow variable is used more frequently as error indicator than its gradient. Indeed, the interpolation and truncation errors are proportional to the second order derivative of the solution [22, 130], and the Hessian matrix is of fundamental importance in anisotropic mesh adaptation to specify a grid spacing according to the directional behavior of the solution [64, 5]. Moreover, as the grid gets more refined near, for instance, a shock, the gradient tends to infinity. In this work, the Hessian matrix $\mathcal{H}(s_i)$ is computed as the gradient of each component of $\boldsymbol{\nabla} s_i$, namely by applying twice Equation (4.2). A scalar error estimator is obtained as the module of the two vectors resulting from the

projection of the Hessian matrix along the directions parallel and perpendicular to the local velocity, respectively, as follows

$$\epsilon_i(s_i) = V_i\sqrt{E(\hat{\boldsymbol{\tau}}, s_i)^2 + E(\hat{\boldsymbol{\zeta}}, s_i)^2} \qquad \text{whit} \qquad E(\mathbf{p}, s_i) = \mathbf{p}^{\mathrm{T}}\mathcal{H}(s_i)\mathbf{p}\,, \tag{4.3}$$

where $\mathbf{p}$ is a generic vector, while $\hat{\boldsymbol{\tau}}$ and $\hat{\boldsymbol{\zeta}}$ are the unit-vectors tangent and normal to the velocity, respectively.

Finally, different error estimators can be mixed together by suitable weights to take into account different flow variables. Consider for instance that, for a certain problem, two different error estimators ϵ_1 and ϵ_2 are supposed to be required to detect all relevant flow features. Therefore, the compound error estimator ϵ_i at each point is built as

$$\epsilon_i = w_1\frac{\epsilon_{1,i}}{\max_{i\in\mathcal{K}}(\epsilon_{1,i})} + w_2\frac{\epsilon_{2,i}}{\max_{i\in\mathcal{K}}(\epsilon_{2,i})}$$

where w_1 and w_2 are two user-defined positive weights, such that $w_1 + w_2 = 1$, that are used to control the relative importance of each single estimator.

4.1.2 Target grid spacing

The target grid spacing is obtained by starting from an initial couple of grid and solution. For each node of the grid, the average length ℓ_i of the connected edges is computed, namely

$$\ell_i = \frac{1}{N_i}\sum_{k\in\mathcal{K}_{i,\neq}} \|\boldsymbol{x}_i - \boldsymbol{x}_k\|$$

where N_i is the number of nodes connected to i by an edge. Then, after the error estimator ϵ_i is computed at each node as described in the previous subsection, the refinement and coarsening thresholds are defined as

$$\begin{aligned} \tau_R &= \mu + k_R\sigma & \qquad \tau_C &= k_C\mu \\ \tau_{R_1} &= \mu + 2k_R\sigma & \qquad \tau_{C_1} &= k_C\mu/2 \end{aligned}$$

where μ and σ are the mean and the standard deviation of the error estimator over the whole domain and k_R and k_C are two user-defined parameters to control the refinement and coarsening, respectively. Finally for each node, the target average length $\bar{\ell}_i$ is defined as

$$\bar{\ell}_i = \begin{cases} 0.25\,\ell_i & \text{if } \epsilon_i \geq \tau_{R_1}\,, \\ 0.5\,\ell_i & \text{if } \epsilon_i \geq \tau_R\,, \\ \ell_i & \text{if } \tau_C < \epsilon_i < \tau_R \\ 2\,\ell_i & \text{if } \epsilon_i \leq \tau_C\,, \\ 4\,\ell_i & \text{if } \epsilon_i \leq \tau_{C_1}\,. \end{cases} \tag{4.4}$$

The choice of defining two thresholds for refinement and two for coarsening, namely $\tau_{R_1} > \tau_R$ and $\tau_{C_1} < \tau_C$, allows a better control of the grid spacing, since where a much larger (lower) error is estimated the average edge length is reduced (increased) more markedly. A final check is performed to avoid excessively small or large edges, i.e. $\bar{\ell}_i = \min(\max(\bar{\ell}_i, \ell_{\min}), \ell_{\max})$.

With the illustrated procedure, a discrete map of desired edge size is obtained, on the basis of which the metric field used to drive grid adaptation is built.

Multi-passage strategy

A possible drawback of the described solution-based adaptation procedure is that, when applied to flow fields characterized by phenomena of different intensities, the grid may be over-refined near the strongest features, whereas the weaker ones may be undetected. For instance, if the flow field encloses both a shock wave and a smooth, continuous rarefaction wave, the grid would probably be refined near the shock, where the error estimator peaks are located, but it would remain unaltered (and so over-coarsened) near the weaker feature.

To circumvent this shortcoming, modifications of the classical error estimates have been proposed in the literature [92, 130]. In this work, the solution proposed by Aftosmis [2], i.e. a multiple evaluation of the error estimation for each adaptation step, is adopted. This technique, called in the following *multi-passage*, consists in re-computing at each passage the refinement and coarsening thresholds excluding the nodes already marked for refinement in the previous passages. In the first passage the thresholds τ_R and τ_C are computed in the standard way. In the next passages, the mean and the standard deviation are evaluated excluding the nodes on which the error is greater than the refinement thresholds computed at the previous passage. Since the mean decreases because of the exclusion of the maximum values, the refinement and coarsening thresholds decrease at every passage allowing to capture also less intense features, as illustrated in Figure 4.1. Finally, the refinement and coarsening thresholds obtained in the last passage are used in Equation (4.4) to compute the target grid spacing, which, thanks to this strategy, presents low values also near the weaker features. At the same time, since the coarsening threshold obtained in the last passage is lower than the initial one, the target edge size is increased for a smaller number of grid points and an excessive coarsening is thus avoided. At the current level of development, the number of passages is a user-defined parameter, which, in the present work, varies between 3 and 5, according to the problem under consideration. In a future development, a control on this parameter may be implemented on the basis of the statistical analysis of the error distribution.

4.1.3 From the grid spacing to the metric map

Metric maps are widely used in mesh adaptation to prescribe the edge length [18, 22, 64, 4, 36, 32, 26]. For a three-dimensional domain Ω, the metric field $\mathcal{M}(\boldsymbol{x})$ is a field of symmetric positive matrices of $\mathbb{R}^{3\times3}$, that defines a Riemannian structure over Ω [52].

The metric field $\mathcal{M}(\boldsymbol{x})$ can be used to control the size of edges. According to the equi-distribution principle, the goal of mesh adaptation is to obtain a unit mesh with respect to this metric, i.e. a mesh that has all edges of unit length [18]. The length of a vector $\boldsymbol{w}$ in the $\mathcal{M}$-metric space is given by

$$\|\boldsymbol{w}\|_{\mathcal{M}} = \sqrt{\boldsymbol{w}^{\mathrm{T}}\mathcal{M}\boldsymbol{w}}\ . \tag{4.5}$$

Now the construction of the metric tensor is described for the isotropic case, i.e. when the desired edge length is the same in all directions. Suppose that in a certain point i of the mesh the desired edge size in the Euclidean space given by Equation (4.4) is $\bar{\ell}_i$. Then, the metric at the point i can be expressed as $\mathcal{M}(\boldsymbol{x}_i) = \bar{\ell}_i^{-2}\mathbb{I}^3$, where $\mathbb{I}^3$ is the 3×3 identity matrix. The metric field $\mathcal{M}(\boldsymbol{x})$ is computed for all nodes in this way and it is stored on a fixed background mesh.

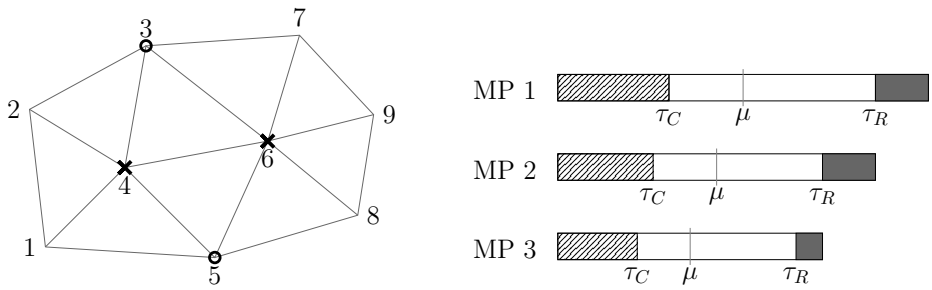

Figure 4.1: Multi-passage strategy application. The rectangles on the right show the variation of τ_R and τ_C due to the 3-passages strategy applied to the grid at left (in 2D for clarity). Since the error estimator values at the nodes 4 and 6 is greater than τ_R computed at first passage (MP 1), the nodes (indicated by ✖) are marked for refinement and excluded from the list of nodes used to computed the thresholds in the next passages. Thus, in the second passage, the mean μ is lower, as is the refinement threshold. These new thresholds lead to the refinement of the nodes 3 and 5, marked by ⭘ . The final thresholds τ_R and τ_C are then computed in the third passage (MP 3) by excluding also these nodes.

During the mesh adaptation process, the length of each edge is computed in the $\mathcal{M}$-metric space to establish if a modification is required. To measure the edge length, an interpolation of the metrics defined at its endpoints is required. In the Mmg3d library [36], for the generic edge e_{ik}, a linear interpolation function is defined as $\bar{\ell}(s) = \bar{\ell}_i + s(\bar{\ell}_k - \bar{\ell}_i)$, with $\bar{\ell}(0) = \bar{\ell}_i$ and $\bar{\ell}(1) = \bar{\ell}_k$. Therefore the length of the edge e_{ik} can be computed as

$$\begin{aligned} \|e_{ik}\|_{\mathcal{M}} &= \int_0^1 \sqrt{\|e_{ik}\|^{\mathrm{T}}\, \bar{\ell}(s)^{-2}\, \mathbb{I}^3\, \|e_{ik}\|}\; ds \\ &= \|e_{ik}\| \int_0^1 \frac{1}{\bar{\ell}(s)} ds = \frac{\|e_{ik}\|}{\bar{\ell}_k - \bar{\ell}_i}\, \ln\left(\frac{\bar{\ell}_k}{\bar{\ell}_i}\right) \end{aligned} \tag{4.6}$$

where $\|e_{ik}\| = e_{ik}^{\mathrm{T}}\, e_{ik}$ is the classical Euclidean edge length, while $\|e_{ik}\|_{\mathcal{M}}$ is the length computed in the $\mathcal{M}$-metric space. To avoid division by zero, if the metrics at the endpoints are equal, the edge length is measure as

$$\|e_{ik}\|_{\mathcal{M}} = \|e_{ik}\| / \bar{\ell}_i\,.$$

In practical cases, as pointed out also in [32], it is usually not possible nor convenient to reach a perfectly unit mesh. Thus, an edge e_{ik} is not modify if

$$\ell_{\mathrm{coll}} \leq \|e_{ik}\|_{\mathcal{M}} \leq \ell_{\mathrm{ref}}$$

where ℓ_{coll} and ℓ_{ref} are two parameters used to control grid adaptation. In this work they are set $\ell_{\mathrm{coll}} = 0.3$ and $\ell_{\mathrm{ref}} = 2.5$.

Anisotropic metric

In isotropic mesh adaptation, the metric map $\mathcal{M}(\boldsymbol{x}_i)$ is actually a scalar matrix that depends on one single parameter ($\bar{\ell}_i$) and it allows to control only the mesh edge size, regardless the edge orientation. Indeed, the geometric locus of all

points P at distance h from a given point O is a sphere centered at O of radius h. Differently, in the anisotropic mesh adaptation the metric map $\mathcal{M}^{\text{an}}(\boldsymbol{x}_i)$ is a general 3×3 symmetric positive definite matrix which allows to control the size, the shape and the orientation of the mesh elements. To better understand how mesh elements are represented within the $\mathcal{M}^{\text{an}}$-metric space, the map is decomposed as $\mathcal{M}^{\text{an}} = R\Lambda R^{-1}$, where R is eigenvector matrix and Λ is the diagonal matrix of the eigenvalues. The distance in terms of the anisotropic map $\mathcal{M}^{\text{an}}$ is still given by Definition (4.5) but, given a point O, the locus of the points P such that $\|P - O\|_{\mathcal{M}^{\text{an}}} = h$ is represented by an ellipsoid, centered at O. The shape of the element is associated to the lengths of the semi-axes of the ellipsoid, which are defined by the eigenvalues Λ, while the orientation (of the ellipsoid and of the element) is defined by the eigenvectors R.

Anisotropic adaptation is not used in the present thesis, because it has not been implemented in the software at the current level of development. However, no particular problems are expected in extending the proposed strategy to the anisotropic case. The main difference concerns the construction of the metric map, which require an error indicator capable to detect the variations of the solution along different direction, like for instance the Hessian operator. Once the anisotropic metric map is built, all the edge lengths can be computed through Definition (4.5) and the proposed approach does not require any further modification.

4.1.4 Extension to different error estimates

The previous sections describe how the metric map is built on the basis of the derivatives of flow variables. This choice of adaptation criteria has been justified by the fact that the present work focuses on the development of a novel adaptive scheme and the choice of the error indicator plays a minor role in the whole approach. Thus, a simple criterion has been preferred to a more accurate one. Nevertheless, the described strategy can be easily extended to different adaptation criteria, in particular an estimate of the numerical error can be straightforwardly implemented.

Thanks to a posteriori analysis, the error due to the numerical approximation can be related to the grid spacing. Mesh adaptation can be used to equi-distribute this error over the mesh by locally modifying the element size, until the desired accuracy is matched [37, 30, 53].

Suppose that u is the exact solution, or a relevant scalar flow variable (e.g. the density), of the considered problem and u_h is the solution obtained solving the discrete governing equations over the mesh $\mathcal{T}_h$. Following [53], an indirect approach can be used to estimate the approximation error, which is defined as $e_I = \|u - u_h\|$. Denoting $\Pi_h u$ the linear interpolate of u on an element of $\mathcal{T}_h$, the approximation error can be bounded by the interpolation error as $e_I \leq c\|u - \Pi_h u\|$, where $c = \frac{d^2}{2(d+1)^2}$ is a constant depending on the spatial dimension d ($c = 9/32$ in 3D).

As shown in [37], the interpolation error over the mesh depends on the Hessian matrix of the variable u. More precisely, on a point $\boldsymbol{x}$ in some neighborhood of the point $\boldsymbol{x}_0$ it can be estimated as $e_I(\boldsymbol{x}) \approx 1/2(\boldsymbol{x} - \boldsymbol{x}_0)^{\mathrm{T}} \mathcal{H}_u(\boldsymbol{x}_0)\,(\boldsymbol{x} - \boldsymbol{x}_0)$.

Hence, it is possible to control the error over the mesh by defining a suitable metric map that relates the edge length to the numerical error. To this end, at

each grid point the metric map $\mathcal{M}(\boldsymbol{x}_i)$ can be defined as $\mathcal{M} = R\widetilde{\Lambda}L$ with

$$\widetilde{\Lambda} = \mathsf{diag}\left\{\min\left(\max\left(\frac{c|\lambda_p|}{\epsilon}, \frac{1}{\ell^2_{\max}}\right), \frac{1}{\ell^2_{\min}}\right)\right\},$$

where λ_p, R and L are respectively the p-th eigenvalue, the matrix of the right and of the left eigenvectors of the Hessian matrix $\mathcal{H}_u(\boldsymbol{x}_i)$ and ϵ is a given threshold for the maximum tolerate error. This metric map can be used to enforce an unit mesh as described in Subsection 4.1.3. As a final remark, it should be noticed that the metric map $\mathcal{M}$ is anisotropic by nature, since the eigenvectors define the principal directions of the flow and the eigenvalues give the associated edge lengths.

4.2 Local mesh adaptation techniques

To efficiently perform three-dimensional mesh adaptation, the capabilities offered by the library Mmg3d, developed by Cécile Dobrzynski and collaborators [36, 35], are exploited. Mmg3d is a fully automatic surface and volume tetrahedral remesher, which iteratively modifies an initial tetrahedral grid to produce an almost unit mesh with respect to the prescribed isotropic metric field.

The software implements a series of different local modifications, like node insertion, node deletion, edge swapping and point relocation, that can be performed both on the interior and on the surface of the grid. Each local modification is performed only if it results in a mesh quality improvement, according to the following quality definition

$$Q_m = \alpha V_m / \left(\sum_{i=1}^{6} \ell_i^2\right)^{3/2}, \tag{4.7}$$

where V_m is the volume of the element, ℓ_i is the edge length and α is a constant parameter introduced to obtain $Q_m = 1$ for a regular tetrahedron, to which corresponds the maximum quality. As the value of Q_m decreases, the quality of the element deteriorates. Indeed, the value $Q_m = 0$ indicates a degenerated tetrahedron [29].

Moreover, to avoid abrupt element size variation over the mesh, the maximum variation that can occur between two adjacent edges is limited by the gradation control technique [19]. Indeed, if the nodes v_i, v_k, v_j are connected by the edges e_{ik} and e_{jk}, it is required that

$$\frac{1}{h_{\text{grad}}} \leq \frac{\|e_{ik}\|}{\|e_{jk}\|} \leq h_{\text{grad}}$$

where h_{grad} is the user-defined gradation parameter.

For what concerns the surface mesh, i.e. the boundary of the computational domain, it can be partitioned into different regions, each represented by a continuous ideal geometry locally reconstructed through cubic Bézier curves. During mesh modification, the geometric criterion of the Hausdorff distance is used to control the gap between the ideal geometry and the actual one [17]. More precisely, a lower value of the maximum allowable Hausdorff distance results in more precise approximation of the curved surfaces, which are indeed discretized by smaller elements.

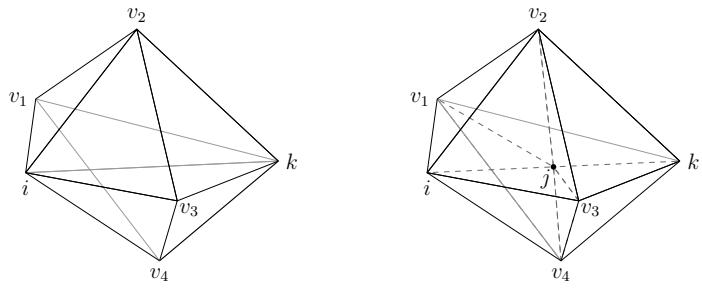

Figure 4.2: Edge split. Left, initial configuration: four tetrahedra share the edge e_{ik}. Right, final configuration: the edge between i and k is split by inserting the new node j in the midpoint, each tetrahedron that had the edge e_{ik} is split in two tetrahedra by the new dashed edges. For instance, the tetrahedron i–k–v_1–v_2 is split in i–j–v_1–v_2 and j–k–v_1–v_2.

The following subsections give a brief overview of the local operators of Mmg3d exploited in the present work to modify the grid. A thorough description can be found in [34, 29, 36].

4.2.1 Node insertion

Three different strategies are used to insert a new node in the triangulation, namely edge split, element split and Delaunay insertion.

Splitting the edge e_{ik} consists in inserting a new point, replacing the initial edge by two new edges and updating the local mesh connectivity. If the edge lies on a boundary face, the position of the new node is computed by means of the Bézier curve representing the involved triangular face, otherwise it is inserted at the middle of the edge. All the elements sharing the edge e_{ik} have to be modified: each of them is divided into two tetrahedra and two new edges are created to connect the new point to the two nodes of the element different from i and k, as shown in Figure 4.2.

Element split is usually performed at the beginning of the adaptation procedure, when two or more faces of a grid element m pertaining to different boundaries, i.e. to different portions of the surface mesh. In this case, to simplify the re-construction of the ideal geometry, the element m is split into four tetrahedra by inserting a new node in its barycenter. As illustrated by Figure 4.3, the four initial faces pertain now to four different elements and each of them can pertain to a different boundary. This operation is performed to split an element initially lying on two or more boundaries, thus element split is performed even if it does not lead to a quality improvement and no minimal angle conditions are enforced. However, since it is performed at the beginning of the mesh adaptation, the shapes of the tetrahedra resulting from an element split can be improved in the successive steps of adaptation.

Finally, also Delaunay triangulation is exploited to insert a new node and to reconstruct the correct topology. Once the position of the new node p is determined according to the metric field, the cavity of the point p has to be identified. This amounts to identifying all tetrahedra having a circumsphere that contains the new point [52]. Labeling r_K^m the radius of the circumsphere of

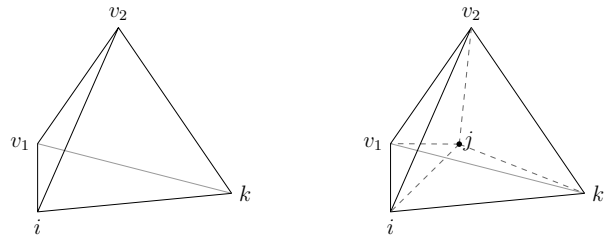

Figure 4.3: Element split. Left, initial configuration. Right, final configuration: the tetrahedron is split by inserting the new point j at its barycenter. Four new tetrahedra are created connecting each original face with the point j.

element m, which is the distance between any vertex of m and its circumcenter O^m, the cavity can be defined as the set of elements for which results

$$\|O^m p\|/r_K^m < 1\,.$$

All the elements of the cavity are then deleted and new tetrahedra are formed by joining the new node to the external faces of its cavity. In order to avoid the creation of very poor quality tetrahedra (the so-called slivers), a procedure of cavity correction is carried out, which consists in explicitly checking that all the resulting tetrahedra have a positive volume. With respect to the simple edge split, the new tetrahedra generated by Delaunay triangulation show usually better qualities. An anisotropic Delaunay triangulation can be easily performed, by computing the lengths r_K^m and $\|O^m p\|$ in the prescribed anisotropic metric field [36].

4.2.2 Node deletion

If an edge results too short with respect to the metric field or to the prescribed minimum edge length, it is collapsed by merging its endpoints into a single one. Suppose that, for the edge e_{ik}, the node i is collapsed onto k. All the elements sharing the edge e_{ik} are removed from the grid and all the other elements sharing the node i are modified and connected to the node k, as shown in Figure 4.4. Despite the simplicity of the deletion procedure, some checks have to be performed before collapsing an edge to assess the fulfillment of quality and validity criteria. If the resulting tetrahedra are not acceptable, namely they have a poor quality or a negative volume, the edge collapse is not carried out. Moreover, for the edges that lie on the surface a check concerning the geometric approximation (Hausdorff distance criterion) is performed.

4.2.3 Edge swap

Edge swap is a powerful technique to increase the mesh quality by modifying only the local connectivity but keeping unchanged the number and the position of the grid nodes. If the edge e_{ik} is selected for swap, the whole element bubble of the edge $\mathcal{B}_{ik}^m$ is re-meshed, namely all the elements sharing the edge. Since a large (combinatorial) number of different configurations are possible, a two-steps procedure is used to simplify the selection of the final configuration, as illustrated

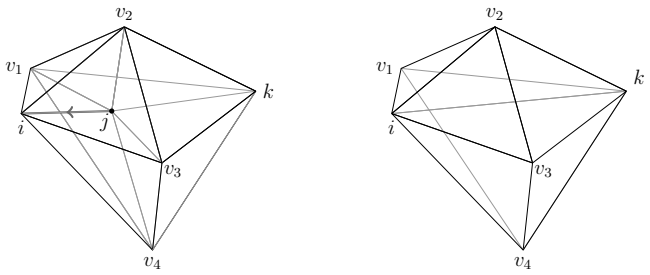

Figure 4.4: Element collapse. Left, initial configuration: element bubble of the node j. Right, final configuration: the edge e_{ij} is collapsed by collapsing the node j onto i. The tetrahedra that shared the edge e_{ij} are deleted, while the ones that shared the edge e_{jk} are modified by substituting the node j with i.

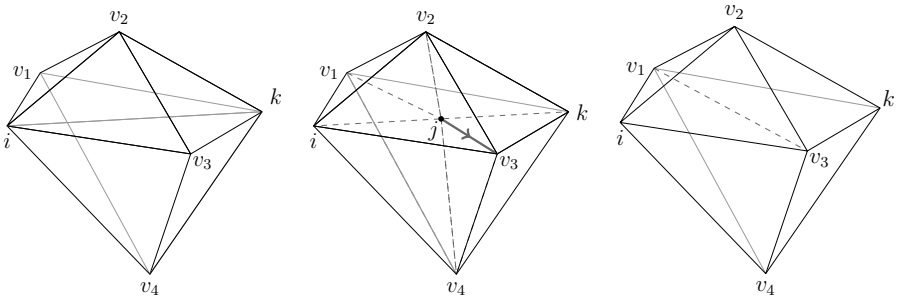

Figure 4.5: Edge swap. Left, initial configuration: four tetrahedra share the edge e_{ik}. Middle, intermediate configuration: the edge e_{ik} is split by inserting the new node j in the midpoint. Right, final configuration: the node j is collapsed on the node v_3 and the new edge between v_1 and v_3 is created. Some tetrahedra created by the split, namely the ones that shared the edge between j and v_3, are deleted while the others are modified by *swapping* the node j with the node v_3.

in Figure 4.5. First, the edge is split at its midpoint and new edges are created to connect the new node to all the other nodes of the bubble $\mathcal{B}_{ik}^m$. Then, a loop on these new edges is started and the collapse of each single edge is tested: if it leads to a valid configuration that increases the mesh quality, it is performed and the swap procedure ends, otherwise the next edge is tested.

A simpler procedure is performed if the edge to be swapped lies on the surface. In this case, the edge is shared by two boundary faces and only one swap configuration is possible. More specifically, the edge shared by the couple of triangular faces is deleted and the other two nodes of the elements are connected by a new edge.

4.2.4 Node relocation

This last operation aims at improving mesh quality by modifying the position of the grid nodes. To this end, a loop on all nodes is performed and the new grid position $\widetilde{\boldsymbol{x}}_i$ for the generic node i is computed. For the internal node i, a barycentric regularization over the elements sharing the node i is performed. The position $\widetilde{\boldsymbol{x}}_i$ is computed as the average of the barycenters of the elements sharing the node i weighted by the cell volume, namely

$$\widetilde{\boldsymbol{x}}_i = \sum_{m \in \mathcal{B}_i^m} V_m \boldsymbol{x}_m \Big/ \sum_{m \in \mathcal{B}_i^m} V_m$$

where $\mathcal{B}_i^m$ is the set of elements sharing the node i, V_m and $\boldsymbol{x}_m$ are the volume and the coordinates of barycenter of the element m, respectively. The node i is moved in the new position $\widetilde{\boldsymbol{x}}_i$ only if the worst quality among the elements of the bubble $\mathcal{B}_i^m$ is improved.

For nodes lying on the surface mesh, the optimal position is computed by considering the ideal geometry [29]. The new position is first computing as the average of the coordinates of the barycenters of the boundary elements sharing the node i projected onto the plane tangent to the point i, then the new position is projected onto the ideal geometry.

4.3 Mesh displacement

This section briefly describes the mesh deformation strategy used when a known movement is imposed to the boundary of the computational domain. According to the ALE formulation presented in Section 2.2, at each time step a suitable mapping of the actual mesh to a new valid one that is conformed to the new boundary is required. This is usually achieved in two-steps: first, the new positions of all boundary points are obtained by a given movement law, then the internal nodes are re-located.

As explained in Subsection 1.2.2, different techniques have been proposed to move the internal grid points keeping fixed the connectivity. In the present work, a mesh displacement strategy is developed on the basis of the elastic analogy proposed by Batina [12], which is easy to be implemented and requires a reasonable computational cost. The mesh is indeed considered as the union of deformable bodies, each of which represents a grid element, subject to a prescribed movement at the boundary. The complete displacement field is obtained by solving

the equilibrium equations for the stress field with Dirichlet boundary conditions expressing the prescribed boundary movement. A standard finite element approach, as described in [13], is used.

Assuming an isotropic elastic material, the stress tensor $\boldsymbol{\sigma}$ can be expressed in terms of the strain tensor $\boldsymbol{\epsilon}$ by the constitutive equation $\boldsymbol{\sigma} = \mathbf{E}\boldsymbol{\epsilon}$, whit

$$\boldsymbol{\sigma} = [\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx}]^{\mathrm{T}}, \quad \boldsymbol{\epsilon} = [\epsilon_{xx}, \epsilon_{yy}, \epsilon_{zz}, \epsilon_{xy}, \epsilon_{yz}, \epsilon_{zx}]^{\mathrm{T}},$$

$$\mathbf{E} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2-\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2-\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2-\nu \end{bmatrix}$$

where ν is the Poisson's ratio, which is chosen as $0 \leq \nu \leq 0.35$ to avoid a bad-conditioning matrix, while E is the Young modulus and it is used to control the stiffness of each mesh element.

Since the smallest elements are usually gathered near the walls or near the relevant flow features, i.e. where an higher accuracy is sought and therefore an higher mesh quality is required, the local element stiffness E_m is set as inversely proportional to element size. More precisely, the following definition is adopted

$$E_m = \frac{1}{\min_{i,k \in \mathcal{K}_e} \|\boldsymbol{x}_i - \boldsymbol{x}_k\|^{\beta}}, \tag{4.8}$$

where $\mathcal{K}_e$ is the set of the edges of the element m and β is a parameter than can be used to control the stiffness variation (in this work $\beta = 1/2$). Thanks to this approach, the largest elements account for the major part of the deformation, while the smallest grid elements move almost rigidly, without undermining their shape.

Lastly, it should be noticed that, in some problems, different type of movements are imposed to adjacent boundaries, like for instance when performing the simulation of a rigidly moving wing connected to the wall at its root. In this cases, the boundary nodes that lie on the wall are constrained to remain on this surface but, at the same time, they have to follow the wing movement. This can be view as a 2D displacement problem in which the role of the internal points is played by the wall nodes while the boundary movement is imposed for the points that lie on the wing root or on the intersection between the wall and a different boundary of the 3D problem. Thus, a bi-dimensional version of the previously described elastic analogy can be applied to obtain the displacement of the wall points, that is then used as boundary condition in the three-dimensional displacement problem.

4.3.1 Robust strategy for large displacements

Robustness is a fundamental property of each mesh deformation strategy, which should be able to deal with significantly large boundary displacements preserving an acceptable grid quality and preventing mesh invalidation, that occurs when an element of the grid has a negative volume. In such a situation, the condition (2.4) is not satisfied and therefore the ALE mapping between the initial and final mesh cannot be defined.

To increase the robustness of the deformation technique, the iterative strategy illustrated in Figure 4.6 is adopted. Suppose to start from the grid $\mathcal{K}^n$, which should be mapped into the new grid $\mathcal{K}^{n+1}$ taking into account a boundary displacement $\boldsymbol{\Delta X}^{n+1}$. The grid $\mathcal{K}^{n+1,*}$ is computed by applying the elastic analogy approach previously described with an imposed boundary displacement equal to $\boldsymbol{\Delta X}^{n+1}$. Then, the validity of the mesh $\mathcal{K}^{n+1,*}$ is assessed by computing the volume of each element. If the test is successful, the computed grid is used as the grid $\mathcal{K}^{n+1}$ and the deformation step is finished. Otherwise, a new grid $\mathcal{K}^{n+\frac{1}{2},*}$ is computed by considering an half boundary displacement, i.e. $\boldsymbol{\Delta X}^{n+\frac{1}{2}} = \boldsymbol{\Delta X}^{n+1}/2$.

If this mesh does not satisfy the validity check either, the boundary displacement is split in half again and the procedure is repeated (d loop) till a valid mesh $\mathcal{K}^{n+\tau,*}$ is obtained or for a certain number of time N_d, which in this work is set as 12. In the first case, the strategy is re-started for the remaining part of the displacement, namely for the interval between $n+\tau$ and $n+1$. In the second case, edge swapping and point-relocation are exploited to restore the grid quality [9, 51] before applying the elastic analogy approach. These mesh adaptation techniques can be efficiently used to eliminate very small and very large angles in grid elements, which most likely lead to negative volumes, without modifying the number of grid nodes, thus without increasing the computational time required by the computation of the solution at the actual time step.

4.4 Summary of the computational procedure

The mesh adaptation strategy described in the previous section of this chapter is now included in the complete computational procedure. For steady simulations this operation is quite straightforward and the resulting iterative procedure consists essentially in three-steps—steady solution computation, error estimate and mesh adaptation—which are repeated till reaching a fixed-point of the pair mesh and solution. On the contrary, different strategies can be adopted to exploit mesh adaptation in unsteady problems, since the grid changes during the time steps. The first option concerns the frequency at which the mesh is adapted, i.e. if mesh adaptation is performed every time level or after a given number of time steps. This choice depends on how large the boundary displacement and the variation of the solution are, but it does not influence the implementation of mesh adaptation phase in the complete computational procedure.

The second, and more relevant, question concerns the solution used to compute the error estimator and thus to drive grid modifications. The standard approach, see for instance [22, 90], consists basically in one steady adaptation step at each time level. For the time interval between t^n and t^{n+1}, it can be summarized as

I. Error estimator computation on the basis of solution u^n.

II. Mesh adaptation to obtain the new grid $\mathcal{K}^{n+1}$.

III. Interpolation of the solution u^n over the new grid, if required.

IV. Computation of the solution u^{n+1} on the grid $\mathcal{K}^{n+1}$, starting from the solution u^n.

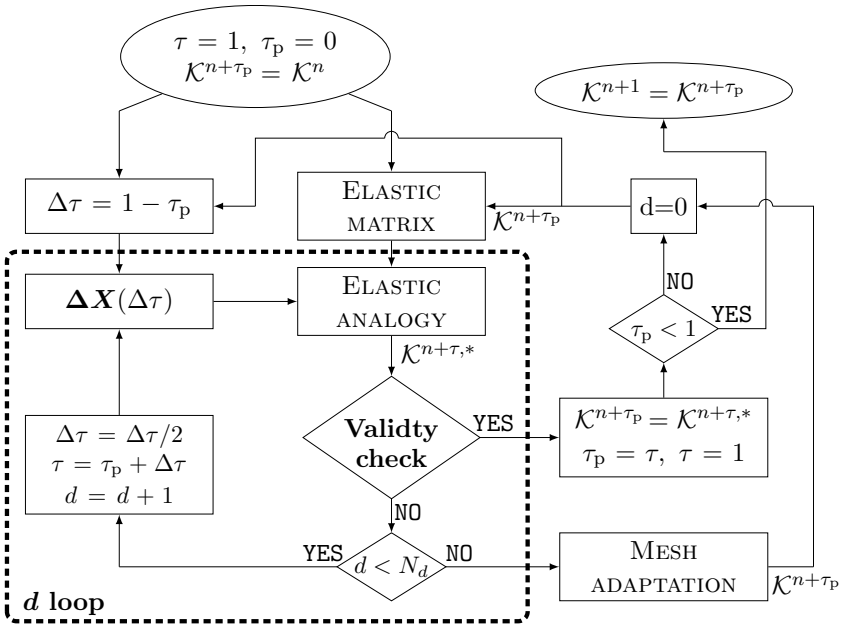

Figure 4.6: Robust strategy for large displacements. Starting from the initial grid $\mathcal{K}^n$, a final grid $\mathcal{K}^{n+1}$ is obtained to comply with the displacement $\boldsymbol{\Delta X}^{n+1}$. A non-dimensional time $0 \leq \tau \leq 1$ is introduced to span the time interval between t^n and t^{n+1}. If the elastic analogy does not lead to a valid mesh, the displacement is reduced, by means of a linear function $\boldsymbol{\Delta X}(\Delta\tau)$ that is defined to express the displacement in terms of τ. If after a certain number N_d of reductions a valid mesh is not found, the mesh quality is increased by mesh adaptation (edge swap and point relocation only) and the procedure restarts with the new mesh $\mathcal{K}^{n+\tau_\mathrm{p}}$. The variable τ_p indicates the percentage of the performed displacement.

This procedure is acceptable only if sufficiently small time steps are adopted, otherwise a significant delay arises between the solution and the mesh adaptation, which is not able to correctly follow the flow features.

A different approach is used in this work to allow larger time steps. At the beginning of the time step, after the mesh has been deformed to comply with the boundary movement, the solution is computed in the so-called *prediction phase*. The unsteady adaptation procedure is now detailed with reference to Figure 4.7. At the beginning of the time step between t^n and t^{n+1}, the grid and the solution are respectively label as $\mathcal{K}^n$ and $\mathsf{u}(t^n, \mathcal{K}^n)$. Then, the following steps are performed:

I. Mesh deformation: the continuous domain is updated to the new position (if required) and consequently the computational grid is deformed. The new grid is labeled $\mathcal{K}^{n+}$ to state that it is an intermediate grid between $\mathcal{K}^n$ and $\mathcal{K}^{n+1}$.

II. Prediction phase: the solution at the new time step, over the grid $\mathcal{K}^{n+}$, is computed.

III. Error estimator computation on the basis of the new solution $\mathsf{u}(t^{n+1}, \mathcal{K}^{n+})$.

IV. Mesh adaptation: the grid $\mathcal{K}^{n+}$ and the metric map computed on the basis of the new solution $\mathcal{M}(\mathsf{u}(t^{n+1}, \mathcal{K}^{n+}))$ are passed as inputs to the library Mmg3d, which performs local grid modifications.

V. Mesh update: the flow solver receives from Mmg3d all performed local modifications $\Delta\mathcal{K}^{n+}$, that are needed to compute the volume swept by the finite volume interfaces through the three-steps procedure describes in Section 3.1.

VI. Computation of the solution u^{n+1} on the grid $\mathcal{K}^{n+1}$, using as initial guess the solution $\mathsf{u}(t^{n+1}, \mathcal{K}^{n+})$.

It should be reminded that, thanks to the ALE interpretation of the local grid modifications as a series of continuous deformations, the interpolation of the old solution over the new grid is not required.

As indicated by *loop s* in Figure 4.7, multiple adaptation cycles can be performed, i.e. the steps III, IV and V can be repeated. In this case, the error estimator is re-computed over the adapted grid $\mathcal{K}_s^{n+}$ and a new metric map $\mathcal{M}(\mathsf{u}(\mathcal{K}_{s+1}^{n+}))$ is passed to Mmg3d. Since the solution is not re-computed at each iteration, few adaptation cycles are usually required to reach a good match between the target grid spacing and the adapted grid $\mathcal{K}_{s+1}^{n+}$.

Of course, if the time step is sufficiently small, the prediction phase can be skipped and the metric map can be built on the basis of the previous solution $\mathsf{u}(\mathcal{K}^n)$, as in the standard approach.

A final remark concerns the computation of the solution, which is performed in a parallel way in order to reduce the computational time of the overall procedure. Thus, before the steps II and VI the grid is partitioned using the library Metis [75] among several processors. Mesh adaptation is performed instead sequentially, then at the end of these steps the complete solution is gathered in one processor only. It should be noticed that, due to grid adaptation, the grid partitioning computed before the step II cannot be used for the computation of the solution

at step VI. Details about the data structure used to handle different grid partitions can be found in [40].

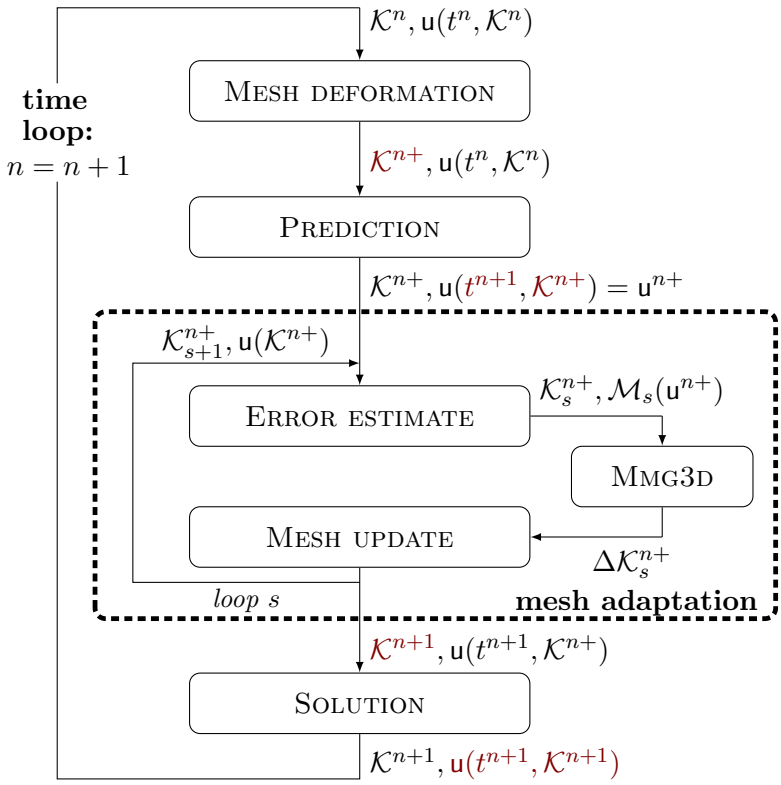

Figure 4.7: Computational procedure for unsteady problems. At the beginning of the time step, the mesh deformation strategy is applied to update the grid $\mathcal{K}^n$ to the grid $\mathcal{K}^{n+}$ that complies with the boundary displacement. A prediction phase is then performed to compute the solution at time t^{n+1} over this new grid. This solution is used to built the metric field $\mathcal{M}(\mathsf{u}(t^{n+1}, \mathcal{K}^{n+})$ that is passed as input to Mmg3d to drive mesh adaptation. The re-mesher communicates to FlowMesh all performed modifications $\Delta\mathcal{K}^{n+}$, so that it can update the finite volumes discretization and compute the swept volumes. The mesh adaptation procedure, highlighted by the thick dashed line, can be repeated (*loop s*). Finally, the solution at the time level t^{n+1} over the adapted grid $\mathcal{K}^{n+1}$ is computed.

Chapter 5

Numerical simulations of reference ideal and non-ideal compressible fluid flows

Simulations of reference compressible fluid flows are carried out to assess the validity of the numerical solution strategy here proposed and implemented. First, bi-dimensional problems are tackled. The results of a regular, steady shock reflection and the unsteady flow field around an oscillating NACA 0012 profile are presented in Subsections 5.1.1 and 5.1.2. Then, the three-dimensional adaptive strategy described in the previous chapter is tested through numerical simulations of a piston-induced shock-tube flow. Finally, two-dimensional simulations of under-expanded nozzle jets are carried out to asses grid adaptation criteria for steady inviscid flows in the proximity of the liquid-vapor-saturation curve, where non-ideal compressible-fluid behavior is expected.

5.1 Ideal-gas reference flows

This first section presents the results of two simulations that assess the validity of the developed numerical tool in two-dimensions. To this end, the flow solver FlowMesh is linked to Mmg2d, the bi-dimensional version of the re-mesher Mmg3d. The importance of these tests is due to the fact that the data structures of the bi-dimensional version of the re-mesher are very similar to those of three-dimensional version. The linking between the flow solver and the re-mesher plays a crucial role in the whole numerical procedure, because the computation of the swept volumes requires a prompt communication from Mmg3d and FlowMesh. Hence, the assessment of the numerical procedure in two dimensions represents a first, fundamental test also for its three-dimensional counterpart.

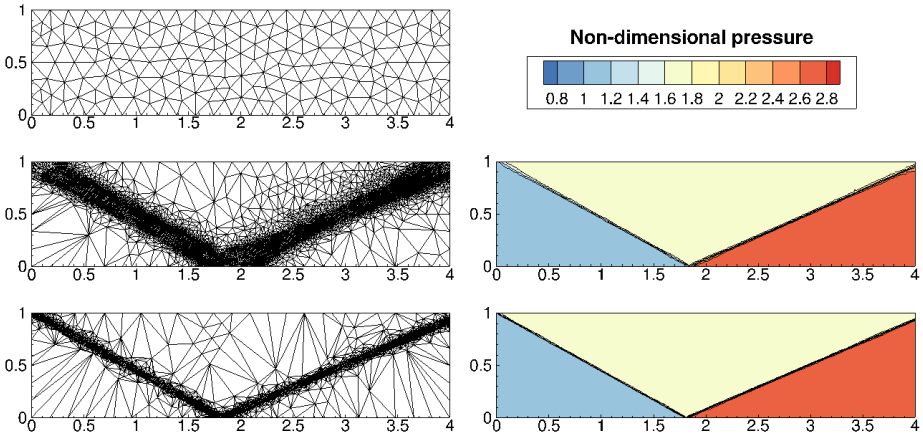

Figure 5.1: Regular shock reflection results: computational grids and pressure contour plots at three different adaptation levels. From top to bottom: original grid (202 nodes, 344 elements), grid and solution after 4 adaptation cycles (9 429 nodes, 18 665 elements), final grid and solution at the 10-th adaptation level (4 092 nodes, 8 087 elements).

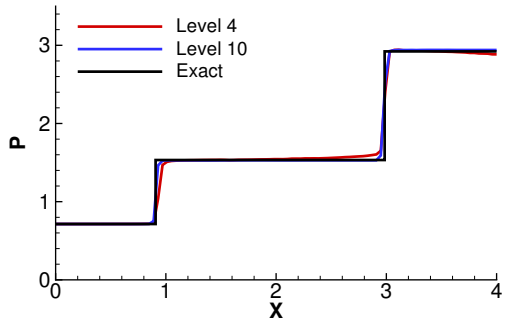

Figure 5.2: Regular shock reflection results: pressure profile at $y = 0.5$. Comparison of the non-dimensional pressure profile obtained after 4 and 10 adaptation levels.

In this section, a steady and an unsteady flow simulations are presented. In Subsection 5.1.1, the results of a regular shock reflection are presented, while in Subsection 5.1.2 describes the numerical investigation of the transonic flow around an oscillating airfoil. In both tests, the polytropic ideal-gas model is adopted and the integrated numerical fluxes are computed through the high-order formulation of Definition (2.30).

5.1.1 Regular shock reflection in dilute gas conditions

The simple, regular reflection of a shock wave over a solid wall is simulated under the ideal gas assumption to verify the performances of the proposed adaptation strategy. The computational domain is a rectangular unstructured mesh with a solid wall at the bottom. The initial solution consists in a uniform flow with at Mach M = 2.377 and incidence $\alpha = -10.95°$. On the left boundary a flow parallel to the wall at Mach M = 2.9 is imposed, while on the top one the initial values

of pressure and density are enforced by means of Riemann boundary conditions. Due to the different boundary conditions, an oblique shock is generated from the top-left angle of the domain, which is reflected at the wall as a shock with different intensity. After the second shock, the flow is again parallel to the wall, but with a different Mach number. Therefore, the flow appears as divided into three separate uniform regions: before the first shock, downstream the reflected shock and between two waves.

The initial grid is a uniform unstructured grid, made of 202 nodes and 344 triangles. The error estimator is based on the Hessian of the Mach number and the refinement threshold is computed with 5 passages to capture both shock waves. Ten complete adaptation cycles are performed. Figure 5.1 shows the evolution of the grid and the solution at different adaptation level. All the variables are made dimensionless by means of the following reference values $P_{\text{ref}} = 1$ atm and $T_{\text{ref}} = 288.15$ K.

As expected, the accuracy of the solution increases with adaptation cycles. A good agreement between the profile of the pressure along the line $y = 0.5$ over the final grid and the exact solution is achieved, as illustrated in Figure 5.2.

5.1.2 Transonic flow around the oscillating NACA 0012 airfoil

Two-dimensional compressible inviscid flow computations around an oscillating NACA 0012 airfoil are presented. The pitching motion of the airfoil is prescribed by the following sinusoidal function

$$\alpha(t) = \alpha_\infty + \alpha_0 \sin(\omega t)\,,$$

with an initial angle of attach $\alpha_\infty = 0.016^\circ$ and an oscillation amplitude of $\alpha_0 = 2.51^\circ$ around the quarter-chord. The oscillating frequency ω is computed so that the reduced frequency is $k = \omega c/u_\infty = 0.1628$, where c is the airfoil chord and u_∞ is the free-stream velocity, as in [83].

A steady transonic simulation of the airfoil at the fixed incidence $\alpha = \alpha_\infty$ and with a free-stream flow at Mach $\mathrm{M}_\infty = 0.755$ is performed first. The initial grid is composed by 1 088 nodes and 1 902 triangles. To enhance the accuracy of the steady solution, an adaption step is carried out and the solution is recomputed on the new grid, made of 2 968 nodes and 5 651 elements.

The steady results are used to start the unsteady adaptive simulation, in which two complete oscillation periods are computed. A Backward Euler scheme then is used for time integration and two different time-steps are tested: Δt_0 and $2\Delta t_0$ obtained by subdividing the whole computational time $T^{\text{sim}} = 4\pi/\omega$ into 200 and 100 time steps, respectively. Defining a non-dimensional reference time as explained in Subsection 2.1.2, one obtains $T^{\text{sim}} = 87.1778$ and $\Delta t_0 = 0.43589$.

Because of the large time-step chosen for the simulations, the solution changes significantly between two consecutive time steps and therefore the adaptation cycle is performed at every time step. The Hessian of the Mach number is chosen as the error estimator and three passages are performed to compute the refinement and coarsening threshold. Figure 5.3 shows the computational grids at different time steps for the simulation performed with Δt_0. The initial grid at the beginning of the unsteady computation is shown at the top-left corner, then the grids at every quarter of the second period are shown in the remaining pictures. A low coarsening threshold is chosen because two-dimensional grids

have a limited number of points with respect to the available computational power and therefore the mesh adaptation process is driven with the primary aim of increasing the solution accuracy disregarding the reduction of grid points.

Figure 5.4 shows the contour lines of the Mach number and the pressure. Thanks to mesh adaptation the shock that moves from the upper to the lower airfoil surfaces is well capture. However, the slip line is not well resolved, probably because the intensity of the discontinuity across the wake is significantly too weak with respect to the intensity of the shock, therefore even the multi-passage strategy is not able to detect it. The choice of different error estimators, based for instance on the derivatives of entropy or vorticity, may possibly allow to capture better this flow feature.

The effect of variation of the time step is finally investigated in terms of lift coefficient. The lift coefficient is an integral quantity and for a wing is defined as

$$C_L(t) = \oint_{\partial\Omega_{\text{wing}}} C_P(\boldsymbol{s}, t)\, n_\infty(\boldsymbol{s}, t)\, d\boldsymbol{s}$$

where $\partial\Omega_{\text{wing}}$ indicated the wing surface and n_∞ is the component normal to the free-stream of the surface outwards normal. Within the node-pair finite-volume formulation here adopted it can be computed as

$$C_L(t) = \sum_{k \in \mathcal{K}_{\partial,\text{wing}}} C_P(\mathsf{u}_k, t)\, \hat{\xi}_{k,\infty}(t)\,.$$

The curve C_L-α obtained with the two different time steps are shown in Figure 5.5 along with the experimental results provided by Landon [83]. The deviation between the numerical and the experimental data is due to the inviscid approximation used in the present work. However, a good agreement with other inviscid numerical results, such for instance those provided by Batina[12], is observed. As expected, the curve C_L-α shows the hysteresis due to the phase lag between the variation of angle of attack and the lift in time [100]. No relevant differences can be observed between the results obtained with different time steps.

5.2 Three-dimensional piston-induced shock-tube flows

The piston-induced shock-tube flow test is carried out to assess the three-dimensional adaptation strategy developed in the present work. A tube of length L with a square section, closed at one end by a wall and at the other one by a piston, initially contains a gas with $\gamma = 1.4$ (e.g. air) at uniform state P_0, T_0, $\boldsymbol{u}_0 = \boldsymbol{0}$. The piston is instantaneously started and moves at constant speed V_P towards the wall. A shock forms at the initial time and it travels through the tube with a constant velocity V_s, see for instance [122]. The resulting flow is therefore separated in two regions: the first one between the shock and the wall, where the fluid is still in quiescent state, the second one between the piston and the shock where the fluid moves at velocity $\boldsymbol{u} = V_P$. When the shock reaches the wall, it is reflected again as a normal shock but with a different intensity and travels back towards the piston with a different speed.

The numerical investigation is carried out starting from the non-dimensional initial conditions $P_0 = 1$ and $\rho_0 = 1$ and imposing the shock wave Mach number

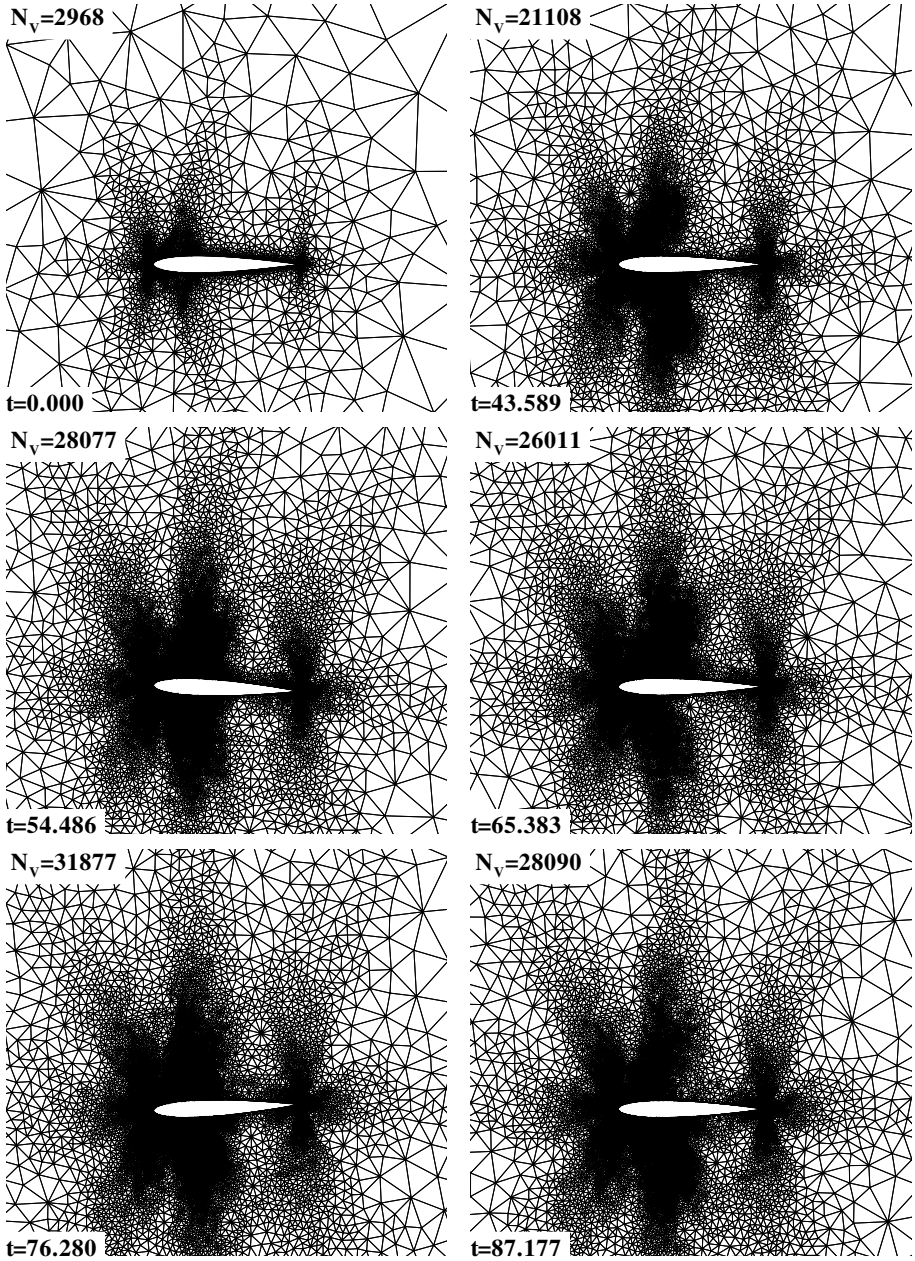

Figure 5.3: Computational grid at different time steps of the unsteady adaptive for the transonic pitching airfoil with $\Delta t = \Delta t_0$. From the top-left to the bottom-right: grid at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\mathrm{sim}}$.

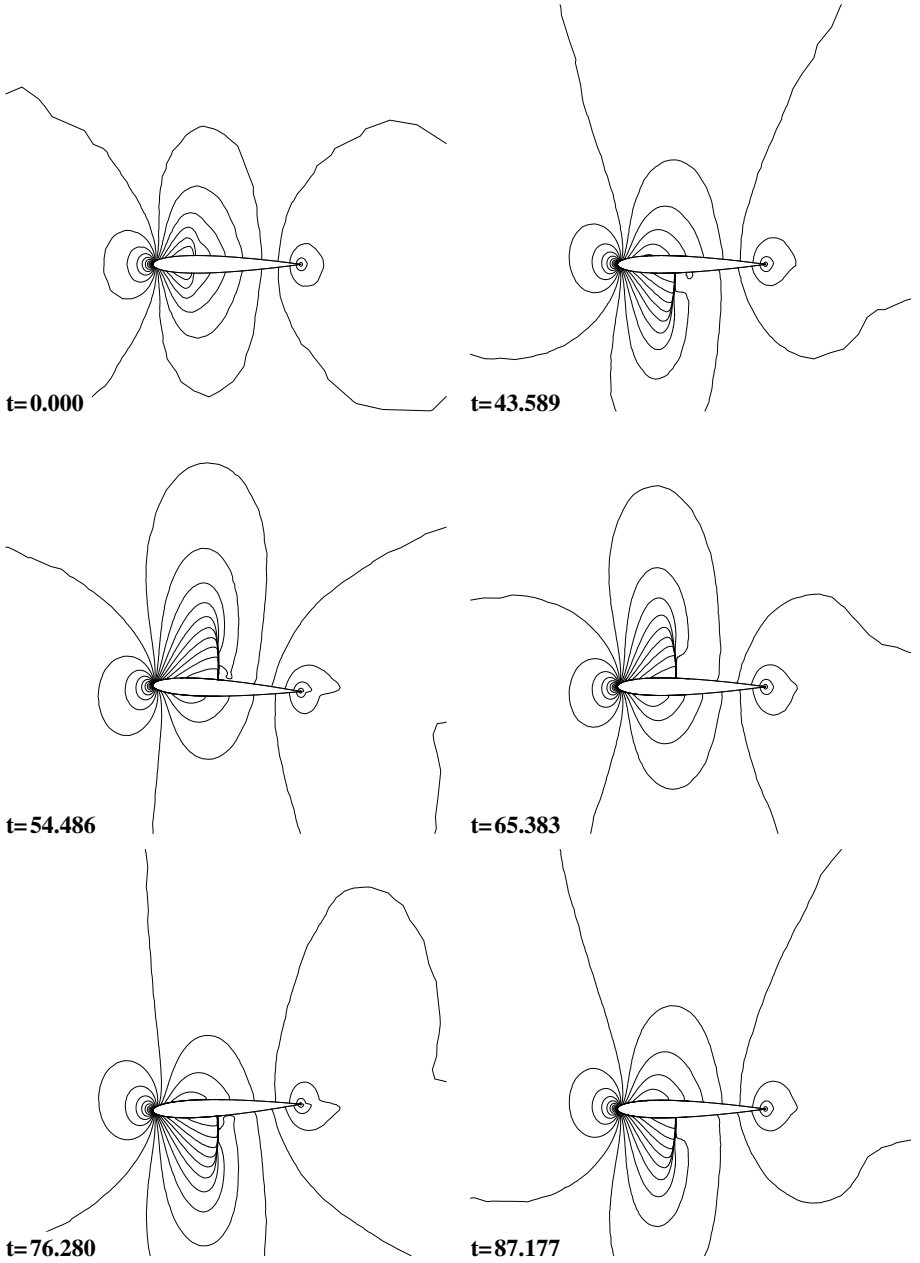

Figure 5.4: Mach contour lines at different time steps of the unsteady adaptive for the transonic pitching airfoil with $\Delta t = \Delta t_0$. From the top-left to the bottom-right: solution at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\text{sim}}$.

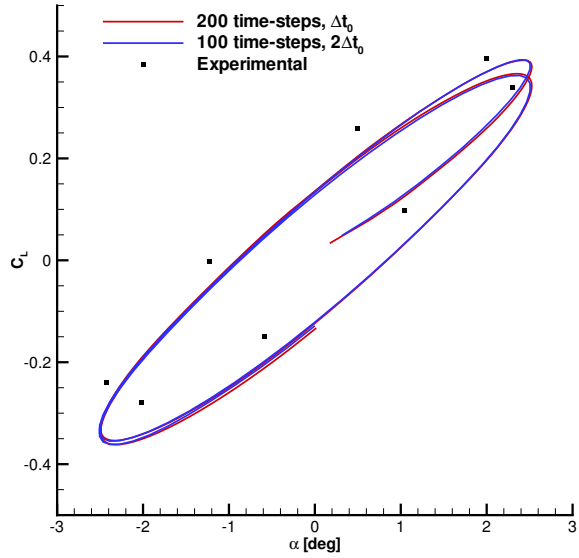

Figure 5.5: Lift coefficient hysteresis of the transonic pitching airfoil. The results obtained with the two different time-steps are shown, along with the experimental results [83].

$\mathrm{M}_s = 2$. As a consequence, the non-dimensional shock speed is $V_s = 2\sqrt{\gamma}$ and the piston velocity is $V_P = 1.25\sqrt{\gamma}$.

The simulation is stopped before the shock wave is reflected by the wall, more precisely when it has performed the displacement $\Delta x_s = 0.9L$, thus at $T^{\mathrm{sim}} = 0.3803$. This time interval is divided into 90 time-steps. The fluid behavior is described by a polytropic ideal gas model. The high-order integrated numerical fluxes and the Backward Euler scheme are used for the spatial and time integration.

The initial computational grid consists in a rectangular box of length L along x and $0.1L$ along y and z. The grid is made by 185 639 tetrahedra and 34 548 nodes. It is shown in Figure 5.6 along with the initial uniform solution. Since the left side (at $x = 0$) represents the piston, during the simulation the computational domain becomes progressively smaller. According to the elastic analogy (see Section 4.3), the major part of this deformation has to be absorbed by the larger grid elements. To facilitate this task, a non-uniform initial spacing is used, whit the smallest elements gather near the piston where a shock is expected to form.

This test case represents a valuable benchmark for the mesh adaptation strategy, because the shock wave moves through the domain and therefore grid refinement is required to correctly capture the shock front, but mesh adaptation is of primary important also to allow such a significant domain deformation, which requires coarsening to avoid extremely small grid elements.

In order to correctly detect the shock wave, the Hessian of the pressure is used as error estimator and mesh adaptation based on solution is performed at each time level. On the contrary, the mesh adaptation step inside the robust deformation strategy for large displacement is performed only when required, as explained in Subsection 4.3.1.

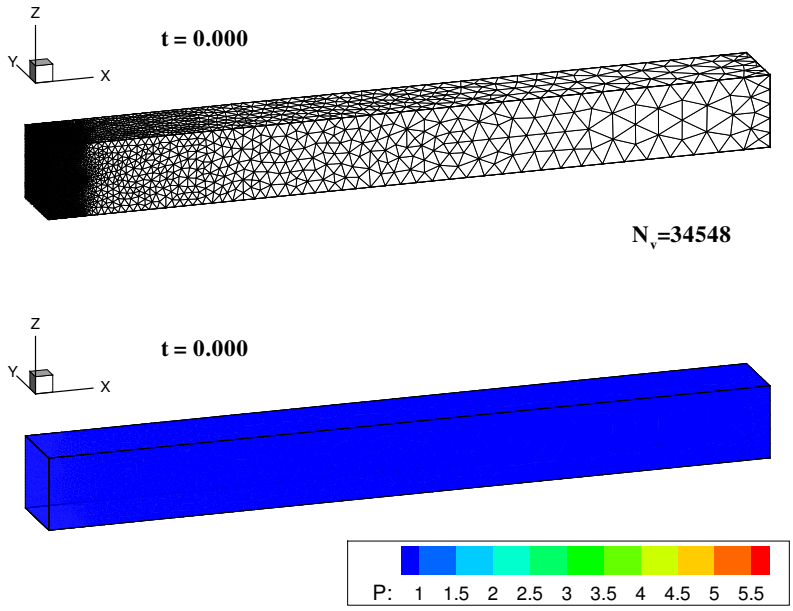

Figure 5.6: Initial mesh and initial uniform solution for piston-induced shock-tube test case, with $P_0 = 1$, $\rho_0 = 1$ and $\boldsymbol{u} = 0$.

Four different combinations of the adaptation parameters and strategies presented in Chapter 4 are tested and they are summarized in Table 5.1. In addition to different coarsening and refinement thresholds, different numbers of multi-passage and adaptation cycles, i.e. s iterations in Figure 4.7, are compared. The profiles obtained at centerline $y = z = 0.05\,L$ obtained with these combinations are shown in Figure 5.7, along with the exact solution. The corresponding grids obtained at $t = 0.211$, i.e. at a displacement of the piston $\Delta x_p = 0.3125$, are shown in Figure 5.8. As expected, the multi-passage strategy and a lower value of the refinement threshold k_R lead to a better detection of the shock wave. To avoid an excessive increase in the number of grid points, a lower value of k_R is combined with an higher value of the coarsening threshold k_C. Comparing test **C** and **D**, it seems preferable a lower refinement threshold than an additional adaptation cycle. Indeed, test **D** provides a solution at the center-line closer to the exact solution than the one obtained in **C**. Furthermore, performing an additional adaptation cycle with the same refinement thresholds—test **C** compared to **A**—does not guarantee an improvement in the solution. This is due to the fact that the solution prediction step is carried out only before the first adaptation cycle, i.e. at the beginning of the adaptation strategy. Then, during mesh adaptation, when a points is inserted, the solution on the new point is retrieved empirically from the nearest old point, on which the solution is computed during the prediction step, and this solution is used to drive mesh adaptation in next adaptation cycle. Unfortunately, if the elements have been already refined many times during the same time step, this solution may deviate significantly from the correct one. Therefore, the mesh modifications in the following adaptation cycle may occur in the wrong positions.

Test	k_R	k_C	Multi-passage	Adapt. Cycles (s)
A	2.8	0.75	3	2
B	2.6	0.85	1	2
C	2.8	0.75	3	3
D	2.4	0.8	3	2

Table 5.1: Combinations of adaptation parameters and strategies used in different test cases for piston-induced shock-tube test case.

Figure 5.7 shows also the results obtained in the test **E**, which is performed using the same adaptation parameters of test **D** but a smaller time step, i.e. the whole time T^{sim} is divided into 360 time-steps. The smaller time step does not lead to a significant improvement in the solution. Moreover, two oscillations appear for the piston displacement $\Delta x_p = 0.5625$, that, as explained in the following, are probably related to the errors generated by the shock-capturing scheme in the computation of the slowly moving shocks.

The grid and the pressure contour at different time steps are shown in Figure 5.9 and Figure 5.10, for the test case **D** only. Mesh refinement allows to correctly follow the shock wave that moves through the domain. Moreover, as it can be noticed for the mesh at time $t = 0.38$, mesh coarsening is correctly applied to the region that has been previously refined to capture the shock once the shock has moved forward.

Finally, Figure 5.11 depicts the pressure, Mach and density profile at the centerline of the tube. A good agreement with the exact solution is reached, as proved by the correct location of the shock wave at the four time steps illustrated. Unfortunately, the shock front is represented over a significant spatial interval, possibly because of a too coarse initial grid. Moreover, two distinct oscillations in the solutions are clearly visible in Figure 5.11 and for test **E** in Figure 5.7. These are more apparent as the grid is refined and are not present during the initial part of the simulations. It is believed that these numerical artifacts are created as the shock wave, which is initially a discontinuity, settles down on its viscous numerical profiles. Indeed, it is well known that in conservative shock capturing schemes disturbances associated to the other characteristic fields are generated as an initial discontinuity is represented over a finite number of grid points/elements [88, 7].

Finer initial grids are not tested, because of the quite high computational time, that for this test lasts for 40 hours on 12 Cores of Intel Xeon. However, this test has assessed the validity of the three-dimensional adaptation strategy proposed in the previous chapter.

5.3 Under-expanded nozzle jets of non-ideal compressible-fluids

Numerical simulations of supersonic under-expanded nozzle jets operating close to the liquid-vapor saturation curve are performed to study the effectiveness of different adaptation criteria in the so-called non-ideal compressible-fluid dynamics (NICFD) region. Since the relationships among the diverse flow variables

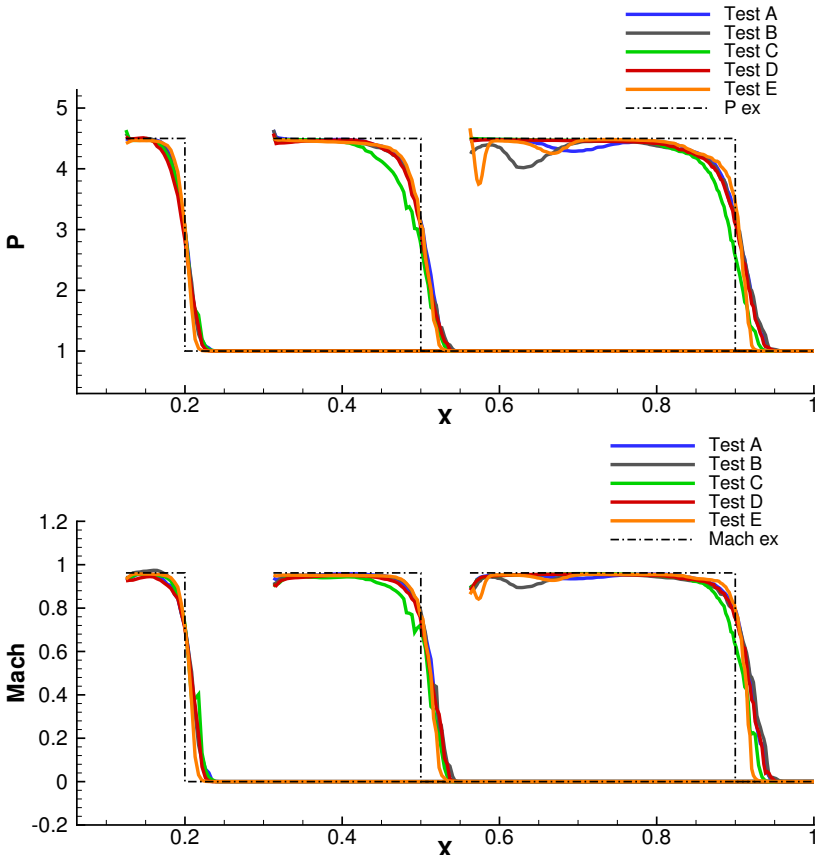

Figure 5.7: Comparison between the solution obtained with different mesh adaptation parameters for the piston-induced shock-tube problem. Centerline pressure (top) and Mach (bottom) profiles obtained at three different times with the combinations of adaptation parameters reported in Table 5.1, along with the test **E** that is performed with the same adaptation parameters of test **D** but with a smaller time step. The piston displacements at the three times are $\Delta x_p = 0.125$, $\Delta x_p = 0.3125$ and $\Delta x_p = 0.5625$, respectively.

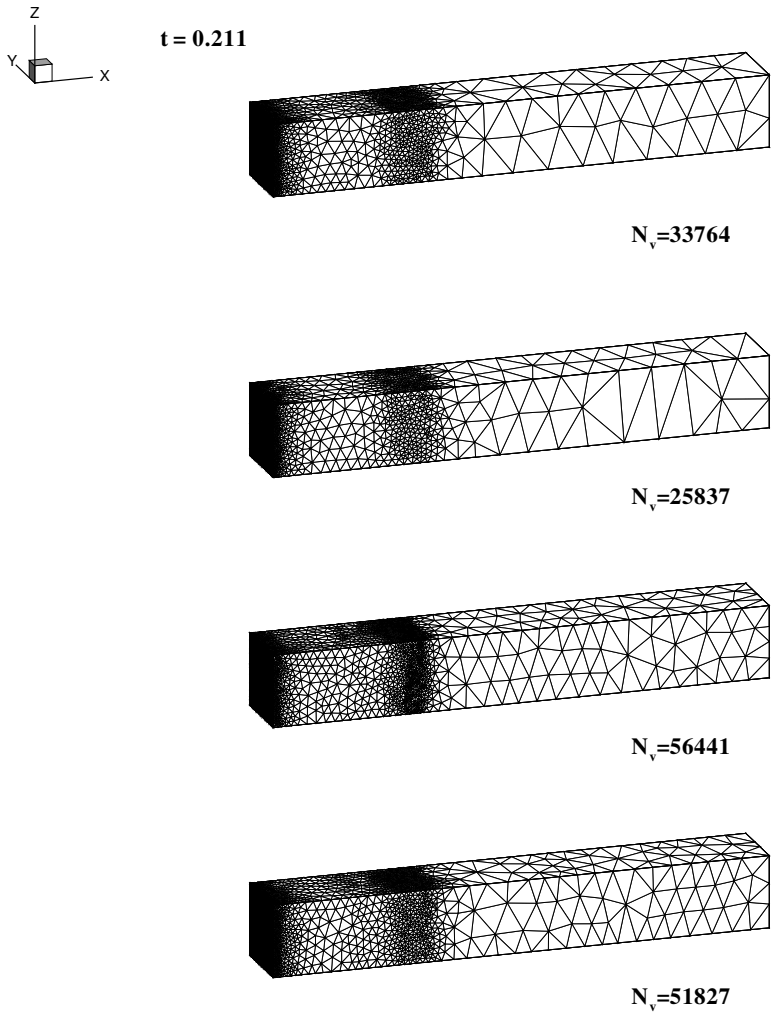

Figure 5.8: Mesh at $t = 0.211$, i.e. at a displacement of the piston $\Delta x_p = 0.3125$. From top to bottom, results obtained in test **A**, **B**,**C** and **D**, according to definitions given in Table 5.1.

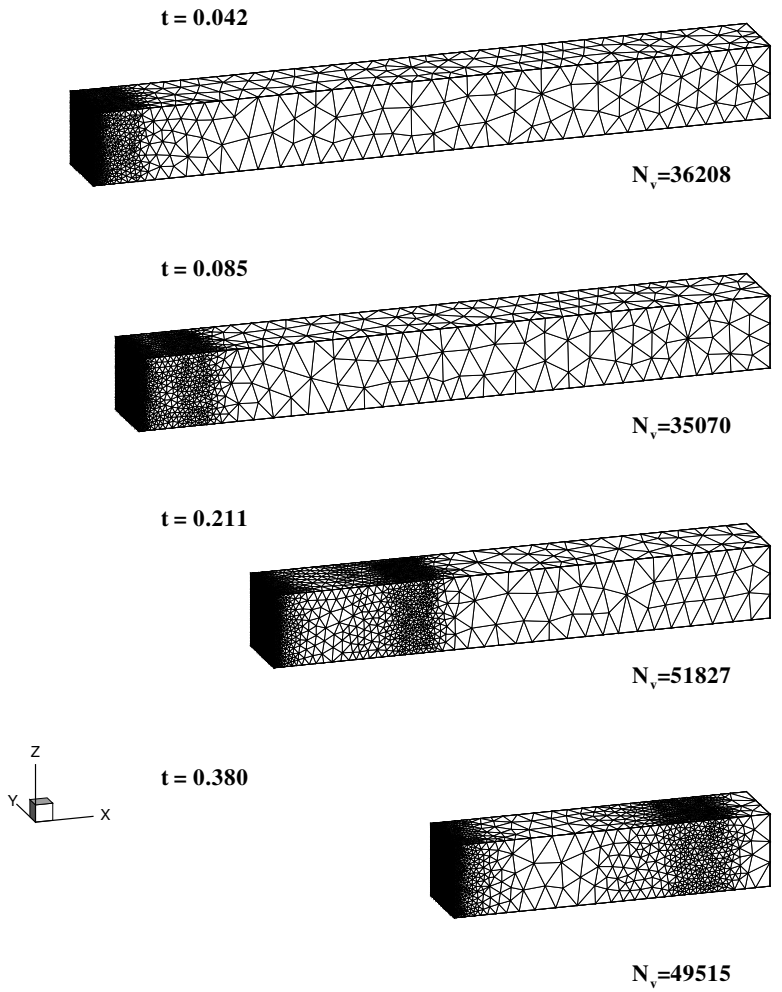

Figure 5.9: Mesh of piston-induced shock-tube problem at different times. From top to bottom, mesh at a displacement of the piston $\Delta x_p = 0.0625$, $\Delta x_p = 0.125$, $\Delta x_p = 0.3125$, $\Delta x_p = 0.5625$.

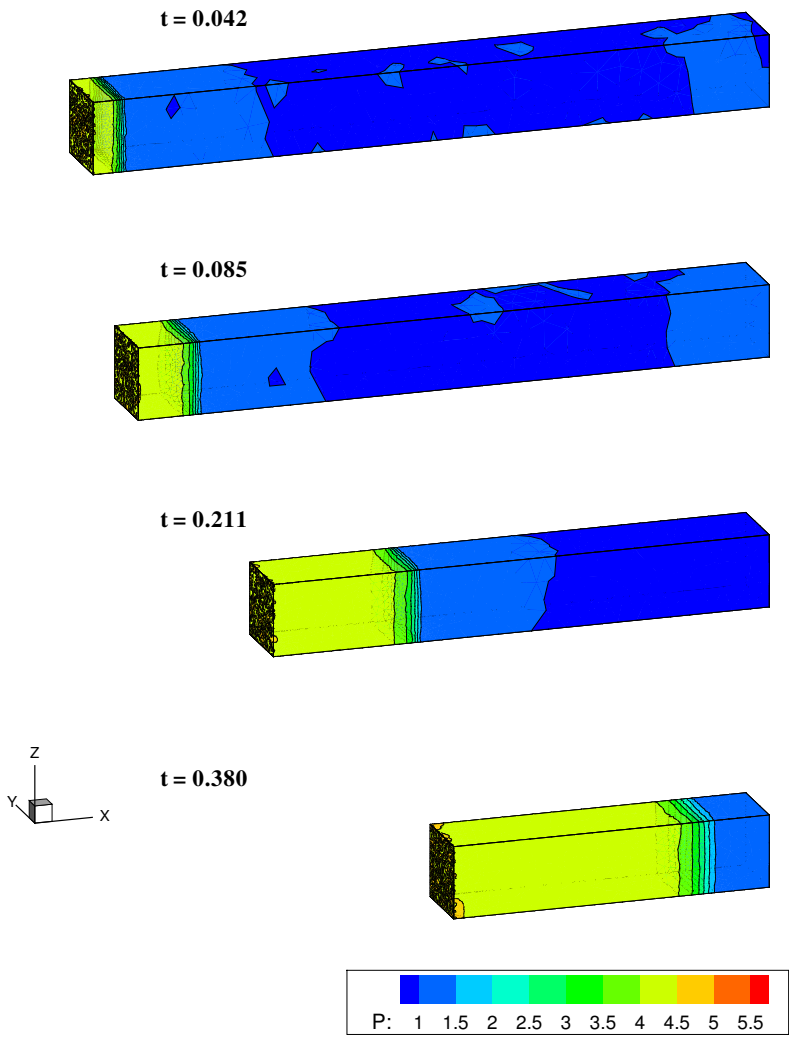

Figure 5.10: Pressure contour plot of piston-induced shock-tube problem at different times. From top to bottom, the contour plot of the pressure at a displacement of the piston $\Delta x_p = 0.0625$, $\Delta x_p = 0.125$, $\Delta x_p = 0.3125$, $\Delta x_p = 0.5625$.

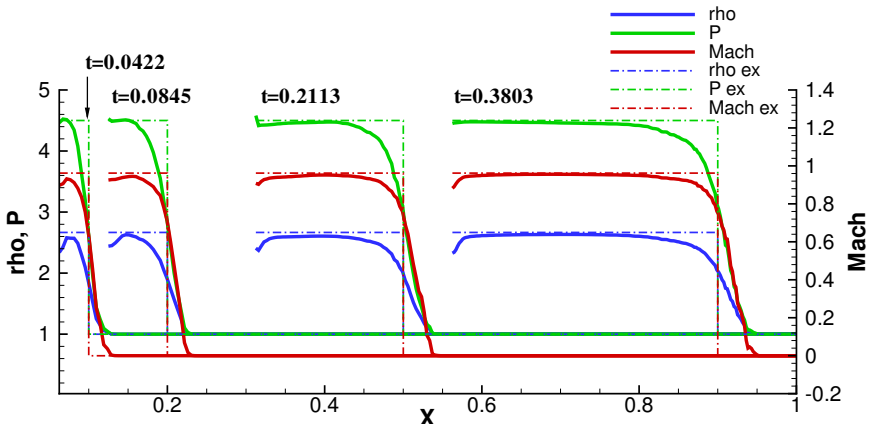

Figure 5.11: Solution of piston-induced shock-tube flow at center-line, i.e. $y = z = 0.05$. Pressure, density and Mach profiles obtained at different times are compared with the exact solution.

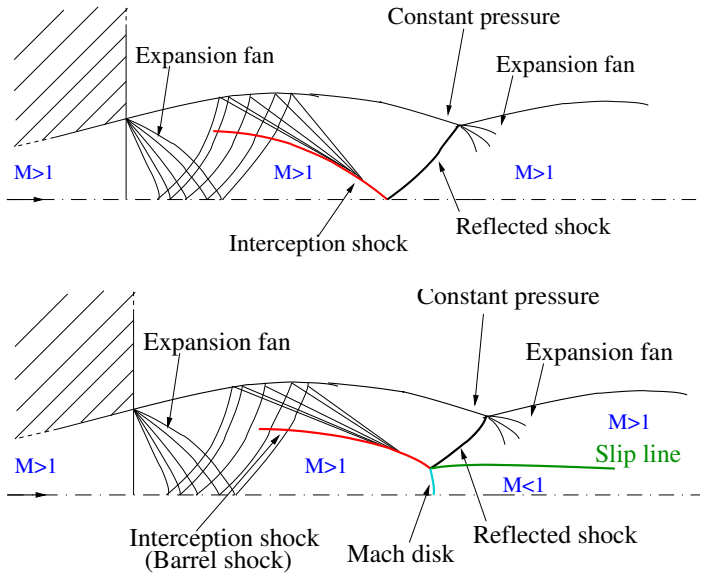

Figure 5.12: Top: weakly under-expanded jet. The interception shock is reflected as an oblique shock and the flow downstream is therefore supersonic. This configuration is referred to as a regular reflection (RR). Bottom: Mach reflection (MR) of a highly under-expanded jet with Mach disk formation. Downstream of the triple point, a slip line separates the subsonic flow near the jet-center (downstream of the normal shock) from the outer region of the flow at supersonic speed.

strongly depend on the assumed thermodynamic model, the selection of the proper error indicator is not straightforward if the fluid thermodynamic departs from the well-known ideal-gas conditions, for which a wide literature regarding the effectiveness of the error indicators is already available [92, 10, 46]. In NICFD, non-ideal thermodynamics results in large compressibility of the fluid, non-ideal dependence of the speed of sound on the density, critical point phenomena, phase transition. Therefore, the applicability of mesh adaptation criteria derived for the constant-specific-heat ideal-gas case is questionable in these highly non-ideal flow conditions. This section presents a first, tentative assessment of the local mesh adaptation criteria for steady two-dimensional NICFD.

Under-expanded jets represent a valuable benchmark for mesh adaptation techniques because of the complex flow structure, depicted in Figure 5.12, which includes oblique and normal shock waves, supersonic expansions and shear discontinuities. A supersonic jets is said to be under-expanded if it exits from a nozzle at a pressure higher than the ambient one. In such a case, the expansion continues outside the nozzle and a Prandtl-Meyer expansion fan forms at the nozzle edge, from which a constant-pressure slip line detaches to separates the jet flow from the quiescent gas. After a first reflection at the jet symmetry axis, the expansion fan is reflected by the slip line as an isentropic compression fan. Propagating downstream, compression waves possibly coalesce into an oblique intercepting shock, which is reflected at the symmetry axis. If the jet is highly under-expanded, i.e. the ratio between the exit pressure and the ambient one is sufficiently large, a normal shock, called Mach disk, occurs and a subsonic region is observed near the centerline. The Mach disk ends at a triple point, where a further slip line develops to separate the supersonic region (behind the reflected oblique shock) from the subsonic one. In axisymmetric flows, the interception shock and the Mach disk form the so-called barrel shock configuration. Following further interaction with the slip line and the symmetry line, a complex flow structure is observed downstream the Mach disk, which influences the shape of the slip line, namely, of the jet shape with respect to the quiescent ambient.

Guardone and co-workers [60] numerically investigated gas non-ideality and caloric imperfection for under-expanded jets of nitrogen gas in supercritical conditions using a finite volume scheme for real gases and modeling the fluid thermodynamics by means of the polytropic van der Waals approximation. Their results in dilute gas regime agree fairly well with the experimental results of Katanoda and co-workers [77, 76]. Following [60], two-dimensional simulations are performed to study the impact of different solution-based adaptation criteria on the solution of under-expanded nozzle jets in NICFD. The van der Waals thermodynamic model is used to take into account, at least qualitatively, non-ideal compressible-fluid effects, which are observed in the proximity of the critical point. The VThermo library, now included in FluidProp [24], has been linked to FlowMesh to compute the fluid thermodynamics.

5.3.1 Setup of numerical simulations

Since the present work aims at investigating the flow structure downstream of the supersonic exit section of the nozzle, only this latter region is simulated. The computation domain is a $30L \times 10L$ rectangular grid composed by triangular elements, where L is the size of the nozzle exit section which is located on the lower

part of the left boundary, i.e. for $x = 0, 0 \leq y \leq 0.5L$. As regards boundary conditions, the nozzle exit section, namely, the inlet section of the computational domain, is represented by a supersonic inlet and the remaining part of the left boundary by a solid wall. A symmetric condition is imposed on the lower boundary, which represents the jet center-line, while constant ambient pressure is enforced along the upper boundary. Non-reflecting boundary conditions are imposed on the right boundary ($x = 30L$) since both supersonic and subsonic flow regions are possibly observed.

The quasi mono-dimensional theory is used to compute the inflow boundary conditions at the nozzle exit, which depend on the stagnation or reservoir state (pressure and temperature) of the gas and the ambient pressure. Eight different operating conditions are considered for nitrogen vapor. The operating conditions are shown in the thermodynamic plane P-v in Figure 5.13 and gathered in Table 5.2, where the subscripts 0, e and a indicate respectively variables in the reservoir, on the exit section and in the ambient. All variables are reduced with respect to their critical values, namely scaled by $T_C = 126.192$ K and $P_C = 3.396$ MPa. The reduced ambient pressure is equal to 0.2 in all cases. To identify the diverse operating conditions, four different values of the ambient density are chosen which identify four different isentropes: the one through $P_a/P_C = 0.2$, $v_a/v_C = 15$ (cases **1** and **2** in Table 5.2) in the dilute gas region and three in the non-ideal compressible-fluid region. Along each curve, two pressure ratios P_e/P_a (3.5 and 5) are investigated. Flow quantities at the exit section, including the Mach number reported in Table 5.2, are computed according to the one-dimensional theory of gasdynamics nozzle, under the assumption of isentropic flows at constant total specific enthalpy [122].

In the following, the results of one operating condition are discussed in details. The selected test is the one indicated as **8** in Table 5.2 because it presents the most significant non-ideal compressible-fluid effects, as revealed by the value of the compressibility factor, which is equal to 0.771 at the exit section.

A reference solution is computed for case **8** over a fine grid of 86 381 nodes and 257 858 elements. In Figure 5.14 the pressure and the Mach contours are shown above and below the symmetry line, respectively. The most significant features of the flow, namely the constant-pressure line which separates the jet from the ambient fluid and the shock reflection which takes place at jet center line around $x/L = 8$, are well captured. Another relevant feature is the interaction between the constant-pressure line and the reflected shock, which occurs very close to the symmetry axis and results in a strong rarefaction wave being reflected towards the axis. Similar flow structures are obtained also in the other operating conditions listed in Table 5.2. Note that the flow downstream of the triple point is not investigated because the viscosity plays a major role in momentum transfer [77], therefore flow features cannot be correctly captured under the inviscid approximation considered here.

The adaptive simulations are started from a low accuracy initial solution, computed using the first-order upwind scheme and a preliminary grid adaptation procedure is carried out on this solution, resulting in a grid composed by 8408 nodes and 24629 triangular elements. This solution is used as the initial condition in the following high-resolution computations.

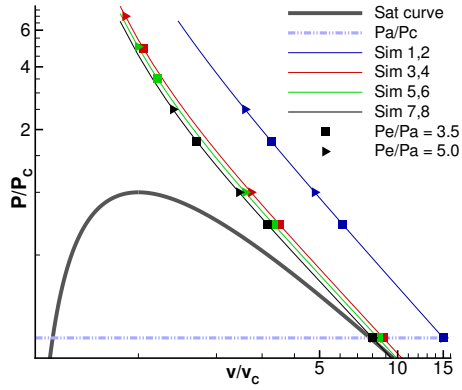

Figure 5.13: Operating conditions of under-expanded nitrogen jets in the thermodynamic plane P/P_C-v/v_C. All variables are scaled with respect to their critical values, i.e. $T_C = 126.192$ K and $P_C = 3.396$ MPa. The nozzle discharges at an ambient pressure $P_a/P_C = 0.2$. Four isentropes, along which the expansions take place, are shown. According to the indexes used in Table 5.2, simulations 1,3,5,7 (indicated with ■) are characterized by a ratio $P_e/P_a = 3.5$, simulations 2,4,6,8 (indicated with ▶) are characterized by a ratio $P_e/P_a = 5.0$.

	Reservoir		Nozzle exit				Ambient	Expansion ratio	
	P_0/P_C	v_0/v_C	P_e/P_C	v_e/v_C	M_e	Z_e	v_a/v_C	P_0/P_a	v_0/v_a
1	1.75	3.266	0.7	6.15	1.220	0.950	15.00	8.75	0.218
2	2.50	2.579	1.0	4.79	1.215	0.950	15.00	12.5	0.172
3	4.90	1.045	0.7	3.50	1.994	0.819	8.80	24.5	0.119
4	7.00	0.890	1.0	2.72	2.010	0.811	8.80	35.0	0.101
5	3.50	1.187	0.7	3.33	1.750	0.801	8.40	17.5	0.141
6	5.00	0.996	1.0	2.58	1.754	0.792	8.40	25.0	0.119
7	1.75	1.683	0.7	3.15	1.237	0.781	8.00	8.75	0.210
8	2.50	1.359	1.0	2.44	1.226	0.771	8.00	12.5	0.170
8id	2.50	1.317	1.0	2.53	1.223	1.000	8.00	12.5	0.165

Table 5.2: Operating conditions considered for under-expanded nitrogen jets.

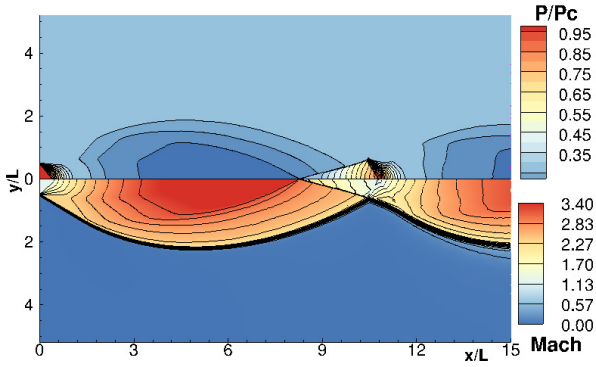

Figure 5.14: Reduced pressure (top) and Mach number (bottom) contour plots of the reference solution of the under-expanded nozzle jet.

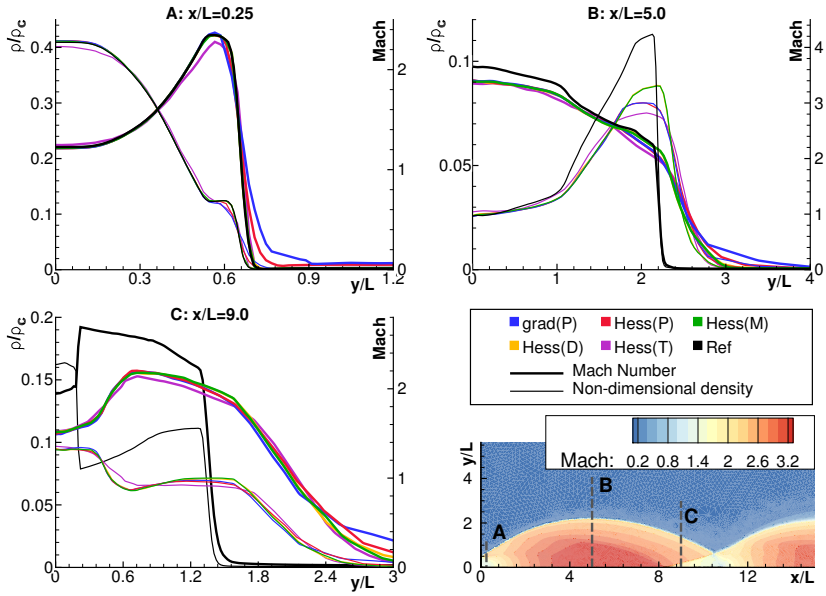

Figure 5.15: Comparison of the results for the under-expanded nozzle jet obtained with different estimators after 3 adaptation steps, using 2 multi-passages. In the legend, grad and Hess indicate respectively the gradient and the Hessian operator, D is the density and Ref is the reference solution. Reduced density and Mach number are shown along three lines at three different positions across the jet center line.

5.3.2 Assessment of mesh adaptation criteria

Three different $x = const.$ sections of the computational domain are selected to compare the density and the Mach number profiles obtained in different simulations. These sections, depicted in Figure 5.15, are located: (A) near the exit section where the Prandtl-Meyer expansion takes place, (B) near the section of maximum width of the jet and (C) across the reflected shock.

The first comparison is carried out by using simple error estimators, based on one variable only. The gradient and the Hessian of pressure, Mach number, density and temperature are tested to drive the adaptation, with $k_R = 2.5$ and $k_C = 0.25$. As shown in Figure 5.15, the Hessian-based estimator is found to perform better than the one based on the gradient, although for simplicity the gradient-based estimator is shown for the pressure only. The expansion fan (section A) is well captured by all estimators, with the only exception of the Hessian of the temperature, which leads to a lower, less accurate maximum value of the density at the end of the expansion. For $x/L = 5$, all the estimators produce an insufficient refinement at the slip line which results in large deviations from reference density profile, probably because of an insufficient number of adaptation steps. Using the Hessian of the Mach number or of the density delivers numerical results that are closer to the reference ones. In all simulations the impact of the estimator on the Mach profiles is less dramatic than on the density. The region of the grid near the reflected shock ($x/L = 9$) is not adequately refined, since the solutions obtained after 3 steps—very similar for all estimators—are very different from the reference one.

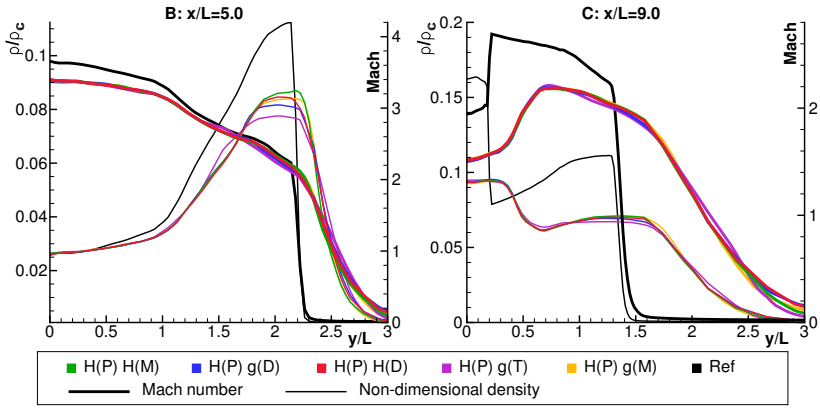

Figure 5.16: Comparison of the results for the under-expanded nozzle jet obtained with different compound estimators after 3 adaptation steps, using 2 multi-passages. In the legend, G and H indicate respectively the gradient and the Hessian operator, D is the density and Ref is the reference solution. Reduced density and Mach number are shown along two lines at two different positions across the jet center line. These positions are the same B and C of Figure 5.15.

Subsequently, different compound error estimators are compared, with $k_R = 2.5$ and $k_C = 0.25$. Figure 5.16 reports a comparison of the density and Mach number across the slip-line and the reflected shock computed by combining the Hessian of the pressure with an additional sensor, which is the Hessian of the Mach, gradient of the density, gradient of the temperature or the gradient of the Mach. The Hessian of the pressure allows to better capture shock waves since the pressure is continuous across the slip line. Based on these results the best estimators appear to be the ones that include the derivatives of the Mach number or of the density. This result agrees with the fluid behavior in NICFD, which is characterized by a large fluid compressibility $\kappa = (\partial\rho/\partial P)_s$. Thus, for the same expansion ratio, larger density gradients are observed in non-ideal compressible-fluid flows with respect to ideal gas ones and so density-based estimators are more suitable for mesh adaptation in NICFD regimes.

As observed also by the previous test, the estimator based also on the temperature under-predicts the density value across the slip line at both positions. In general the use of compound estimators produces results that are in better agreement with the reference. Tests including the temperature as third variables in the error estimator have been performed but no significant improvement was observed. Unfortunately, neither the constant-pressure line nor the reflected shock are sufficiently defined and more adaptation steps or multi-passages are therefore required.

To well capture the reflected shock (at $x/L = 9$), which has an intensity lower than interception shock, the multi-passage strategy is exploited and 5 passages are proved to be sufficient to detect also this feature. As regards the constant-pressure line, it should be pointed out that, because of linearity of the associated characteristic field, slip lines are extremely hard to capture using an artificial viscosity numerical scheme, such as the Roe scheme used here to capture non-linear shock waves. To this end, grid adaptation procedure is of primary importance to

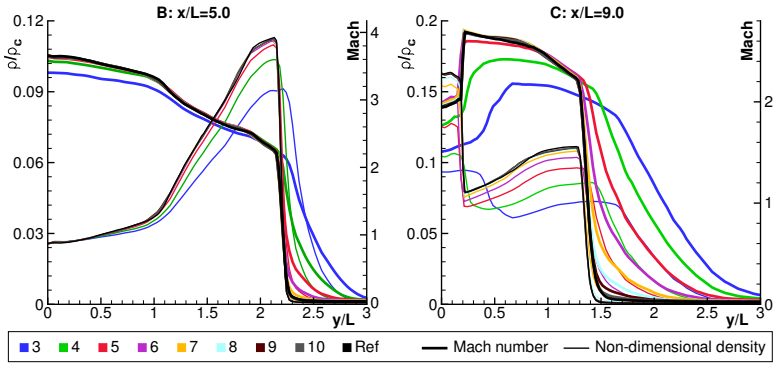

Figure 5.17: Solutions of the under-expanded nozzle jet obtained using the estimator $\mathcal{H}(P)+\mathcal{H}(\rho)$ and 5 multi-passages at different adaptation steps (from 3 to 10). Reduced density and Mach number are shown along two lines at two different positions across the jet center line. These positions are the same B and C of Figure 5.15.

attain an acceptable level of spatial resolution, thus reducing the effect of numerical viscosity adequately. Figure 5.17 shows how the solution varies by increasing the number of adaptation steps (from 3 to 10), using 5 multi-passages and the error estimator $\mathcal{H}(P) + \mathcal{H}(\rho)$. Six steps proved to be sufficient to obtain convergence of the adaptation procedure, i.e. to obtain a computational grid which satisfies the imposed level of error uniformity (k_C and k_R). The number of grid nodes after six adaptation steps is 41 644 nodes. A comparison of the grid and the Mach contour plot after 1 and 6 adaptation steps is shown in Figure 5.18.

Furthermore, Figure 5.19 shows the number of grid nodes and elements at each adaptation level for the estimator $\mathcal{H}(P) + \mathcal{H}(\rho)$ using 2 and 5 multi-passages. If only 2 passages are performed, the number of grid nodes reaches a maximum value after five steps (14 222 nodes, 27 797 elements). In the following steps, the number of removed nodes is greater then the new ones. When 5 passages are carried out, the number of nodes increases more rapidly during the first five steps, then it follows still an increasing trend but less steep.

Finally, a comparison to the ideal gas model results was performed. The boundary conditions at the nozzle exit section were computed imposing the same ambient conditions, pressure ratios P_e/P_a and P_0/P_e and reservoir pressure of the real-gas case **8**. The numerical values are displayed in the last row of Table 5.2. Figure 5.20 compares the solution computed with van der Waals gas model and ideal gas model after six adaptation steps and using the error estimator $\mathcal{H}(P) + \nabla(\rho)$. As expected, the ideal gas model shows a different location of the shock reflection and, in particular, the reflected shock is downstream the one predicted by the reference solution. Furthermore, the same combination of adaptation parameters applied to the ideal gas case results in a different and less refined grid with respect to the van der Waals model (17 229 nodes).

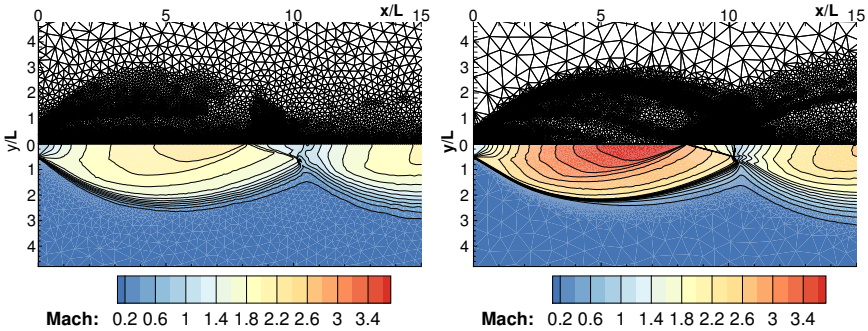

Figure 5.18: Grid and Mach contour plot of the under-expanded nozzle jet obtained using the estimator $\mathcal{H}(P) + \mathcal{H}(\rho)$ and 5 multi-passages after 1 and 6 adaptation levels.

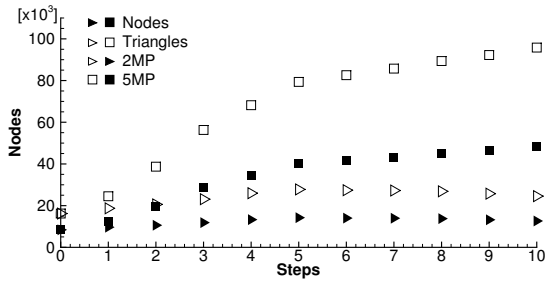

Figure 5.19: Increase of grid nodes and elements with adaptation steps using 2 and 5 multi-passages, and the estimator $\mathcal{H}(P) + \mathcal{H}(\rho)$.

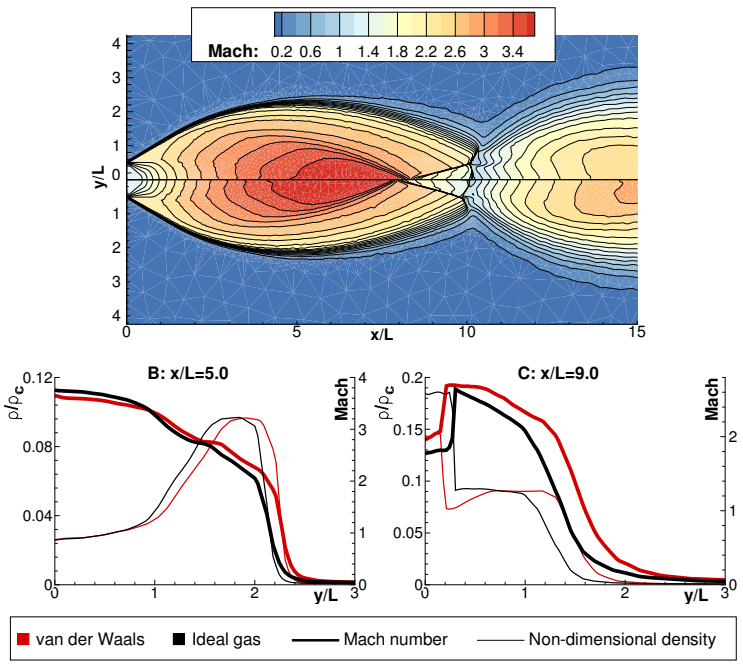

Figure 5.20: Solution of the under-expanded nozzle jet obtained using the estimator $\mathcal{H}(P) + \nabla(\rho)$ and 5 multi-passages with van der Waals gas model and ideal gas model. Top: Mach contour plot, with van der Waals model above the center-line and the ideal gas model below. Bottom: reduced density and Mach number along two lines at the section B and C of Figure 5.15.

Chapter 6

Three-dimensional flows around wings

The proposed three-dimensional conservative adaptive scheme is here assessed for problems of aeronautical interest, by performing numerical simulations of standard test cases involving flows around wings. Section 6.1 presents a steady computation of transonic flow around the ONERA M6 wing, during which mesh adaptation is exploited to achieve a grid spacing suitable to resolve the peculiar lambda shock. An oscillating infinite-span NACA 0012 wing is then investigated in Section 6.2 and the three-dimensional results are compared with the bi-dimensional ones obtained in the previous chapter. To assess the capability of dealing with arbitrary large movements, an unsteady simulation of the wing traveling through the still domain at the flight velocity (laboratory reference frame) is performed. Finally, similar simulations are carried out also for a finite-span NACA 0012 wing, a model which is of primary importance in rotor-crafts CFD since it can represent the tip of a rotor blade.

6.1 Transonic steady flow around ONERA M6 wing

The popular test case of the transonic flow around the ONERA M6 wing is tackled to assess the capability of the conservative adaptive scheme to correctly capture the shock wave that occurs over three-dimensional wing in standard flight conditions. The inviscid solution is computed for a free-stream Mach $\mathrm{M}_\infty = 0.8395$ and an angle of attack $\alpha = 3.06°$, with an ambient pressure $P_{\mathrm{ref}} = 3.11866$ atm and a temperature $T_{\mathrm{ref}} = 255.6$ K. In such flight conditions, the well-known lambda-shock appears on the upper surface of the wing. As explained by Kuzmin [80], this peculiar configuration is due to the swept leading edge of the wing and to a small curvature of the ONERA D profile, which is used to build the M6 wing. The polytropic ideal gas model is adopted to describe fluid behavior and the high-order formulation is used to computed the integrated numerical fluxes.

The computational domain consists in a rectangular box, in which the root plane is characterized by $z = 0$, the opposite plane by $z = 9.2c$, the lateral planes by $x = \pm 16c$ and the upper and lower planes by $y = \pm 14c$, where c is the wing chord at the root, with $x = 0$ at the leading edge and $x = c$ at the trailing edge. The wing span is approximately $b = 1.48c$.

The initial mesh contains 93 432 nodes and 517 364 tetrahedra, with a quite uniform spacing over the wing surface. Six complete cycles of mesh adaptation and solution computation are performed. The grids and the Mach contour plots at different adaptation cycles are shown in Figure 6.1. The peculiar lambda-configuration formed by the Mach iso-lines becomes more and more definite as more adaptation cycles are performed. Starting from a relatively coarse grid, the coarsening strategy is exploited only in the last two cycles, and, even in these cycles, a weak coarsening threshold is used. Thus, the number of grid points progressively increases: the mesh contains 341 481 nodes after 3 adaptation cycles and 626 227 nodes after six. However, it should be noticed that the adaptation strategy is capable of identifying precisely where the refinement is required and the grid spacing is reduced in a quite limited region of the grid.

The profile of the pressure coefficients obtained on the final grid at six different sections along the wing span are shown in Figure 6.2, along with the experimental data [114]. The deviation between the converged numerical results on the adapted grid and the experimental results can possibly be justified by the inviscid approximation adopted in the present work. Nevertheless, a good agreement in the central part of the wing and quite far from the shock is reached.

To evaluated the solution improvement achieved thanks to mesh adaptation, a numerical simulation on a fixed fine grid is performed. Figure 6.3 displays the fixed computational grid, which is composed by 525 688 nodes and 3 065 274 elements, and the solution. The lambda-shock appears to be less resolved with respect to the solution obtained after six and even after three adaptation levels. The improvement in the solution appears more clearly in Figure 6.4 which shows the pressure coefficient profiles at $z/b = 0.65$ obtained with different adaptation cycles and over the fixed fine grid. It can be noticed that three adaptation steps are sufficient to reach a better accuracy with respect to the fixed grid, despite a reduction of about 65% of the grid points.

A final remark concerns the computational time. The whole adaptive simulation (six mesh adaptation steps plus six solution computations) takes $8d - 17h : 10m$ on 12 Cores of Intel Xeon CPU X5650, 2.67 GHz. Such an high computational time is due to the fact the software tool is still under-development and it is not optimized, and only a rough parallelization of the solution computation procedure is available, see Subsection 4.4. Nevertheless, a comparison between the fixed-grid computation and a single adaptation step can give a rough estimate of the overhead due to adaptation. The fifth adaptation step is chosen for this comparison, because the number of grid nodes increases from 470 428 to 558 683 and so it is comparable with the number of nodes of the fixed grid. On the same 12 Cores, the fixed computation takes $29h - 29m$ while the adaptation step takes $38h - 2m$, so an overhead of about 27% occurs, but an higher accuracy is reached.

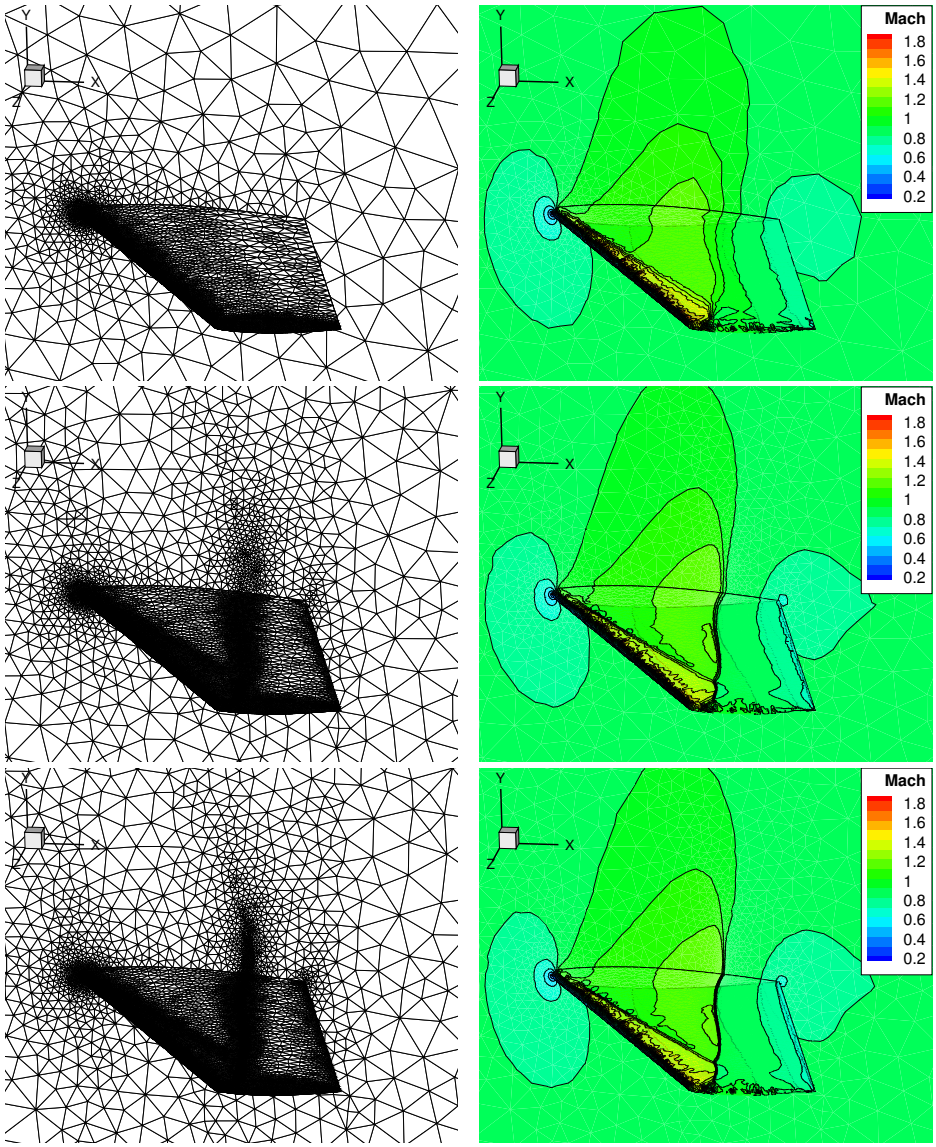

Figure 6.1: Grid and Mach contour plots of the transonic ONERA M6 flow after different adaptation cycles. In the first line, initial grid and solution; in the second and third line, grid and solution after 3 and 6 adaptation cycles.

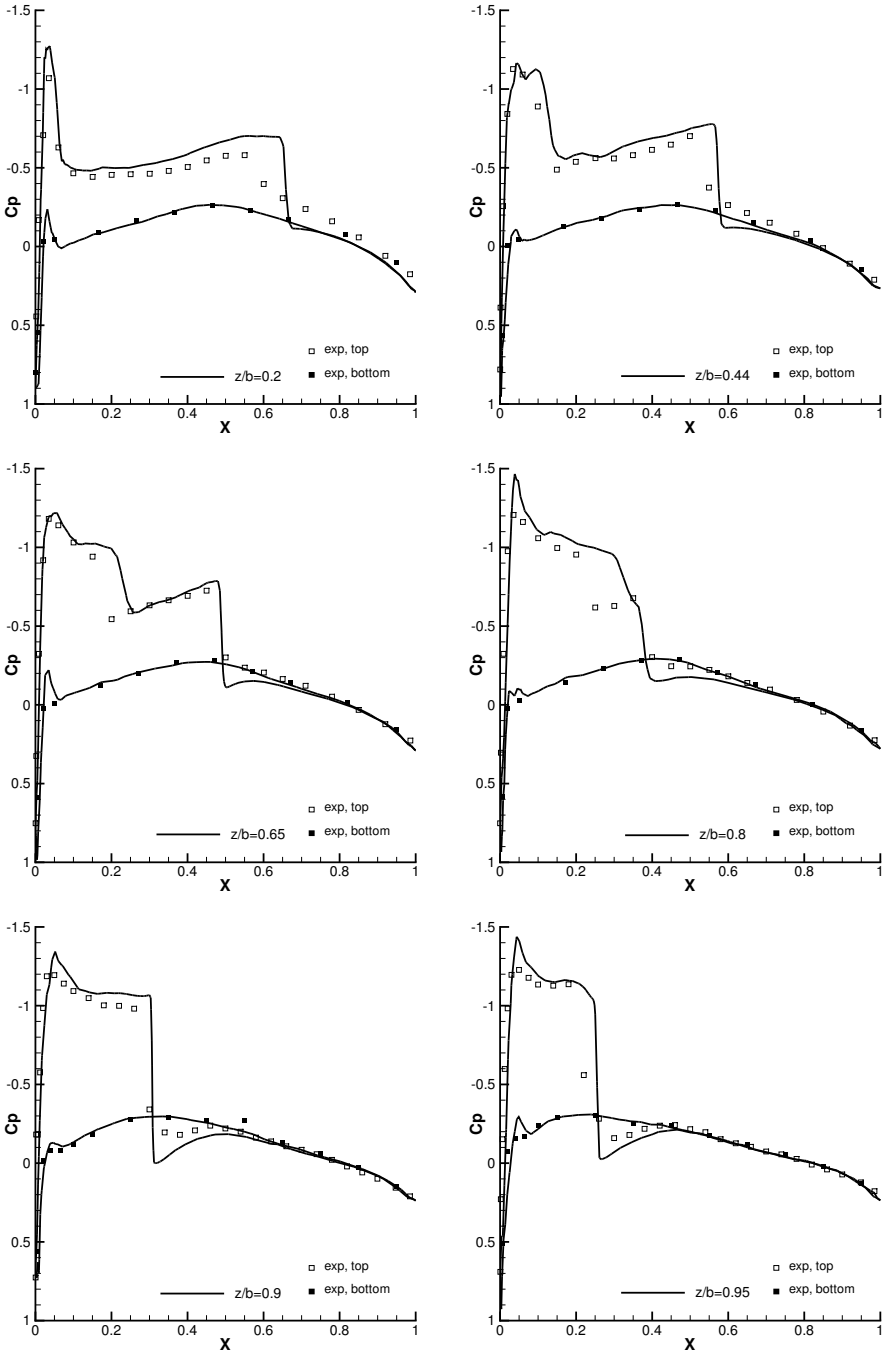

Figure 6.2: Pressure coefficients at different sections of the ONERA M6 wing at $M_\infty = 0.8395$ and $\alpha = 3.06°$. Comparison between the numerical results obtained after six adaptation steps and the experimental ones [114].

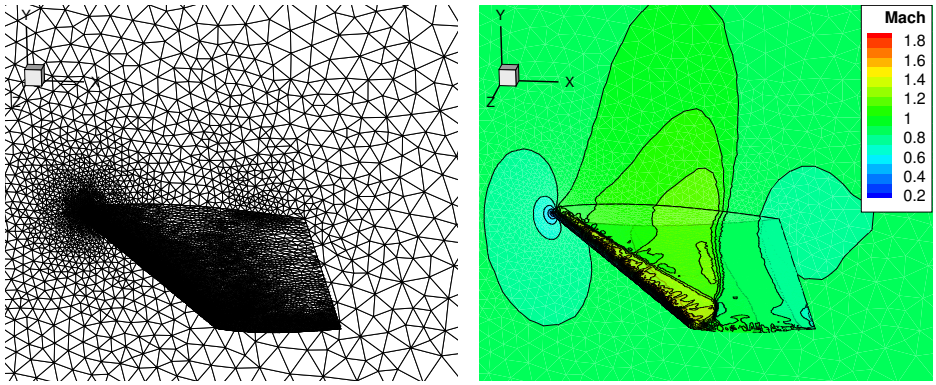

Figure 6.3: Fixed fine grid and solution for the ONERA M6 wing at $M_\infty = 0.8395$ and $\alpha = 3.06^\circ$.

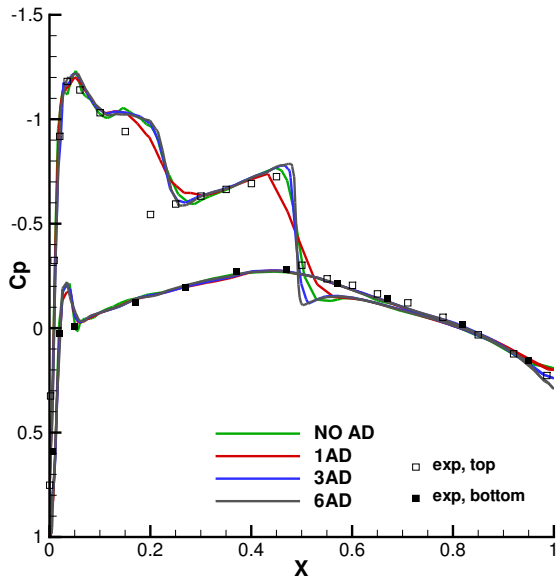

Figure 6.4: Pressure coefficients at $z/b = 0.65$ of the ONERA M6 wing at $M_\infty = 0.8395$ and $\alpha = 3.06^\circ$ obtained over the fixed fine grid and after a 1, 3 and 6 adaptation cycles.

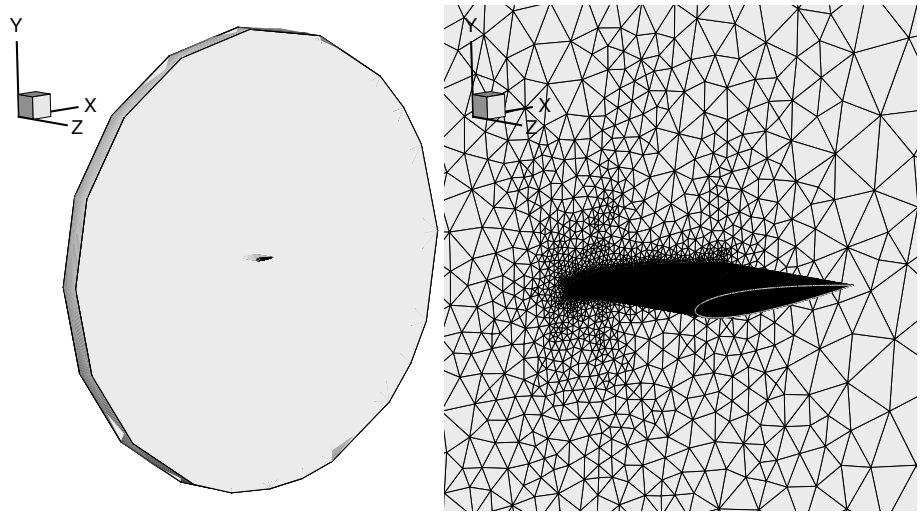

Figure 6.5: Geometry of the computational domain for the infinite-span NACA 0012 wing test. The right picture shows a detail of the region near the wing.

6.2 Infinite-span NACA 0012 wing

Numerical simulations of the infinite-span NACA 0012 wing are performed and compared to the bi-dimensional results. The flow around an infinite-span wing is indeed bi-dimensional since it varies only in the x and y directions and the behavior of the flow around each section along the span (z direction) is equal to the one around the airfoil.

The computational domain is built as shown in Figure 6.5. The wing span is equal to its chord c and the far-field consists in a cylindrical surface with a radius of $12c$, centered at the leading edge. To model the infinite span, the lateral planes at $z = 0$ and $z = c$ are considered as walls, which, under the considered inviscid approximation, corresponds to a symmetry boundary condition.

In all the numerical experiments presented in this section the flow behavior is modeled under the polytropic ideal gas assumption and the high-order formulation is used to computed the integrated numerical fluxes. For unsteady simulations, the Backward Euler scheme is chosen for the time integration.

6.2.1 Steady simulation in the wing reference

A steady simulation of the infinite-span NACA 0012 wing at M = 0.755 and $\alpha_\infty = 0.016°$ is performed and used as the initial condition for the unsteady simulations presented in the next subsections.

A quite coarse grid made of 26 142 nodes and 94 561 triangles is used to start the simulation. Since the domain geometry is re-constructed by Mmg3d from the initial grid, very small elements are gathered on the wing surface to allow an accurate representation of the wing geometry. To increase the accuracy of the solution computed on this grid, two adaptation steps are performed and the grids and the solutions of Figure 6.6 are obtained. The final grid contains 105 002 nodes and 555 229 tetrahedra. Figure 6.7 shows the details of the final

grid near the trailing and leading edge.

6.2.2 Transonic flow around the oscillating NACA 0012 wing

The same unsteady simulation of the transonic flow around a pitching NACA 0012 airfoil of Subsection 5.1.2 is performed over the infinite-span wing. Therefore, the wing experiences an oscillation with amplitude $\alpha = \alpha_\infty + \sin(\omega t)$ around the line connecting the quarter-chord of all sections. The external boundary representing the far-field is kept fixed, while the mesh on the boundaries representing the walls are deformed to allow the movement of the wing roots.

A combined error estimator is used to drive grid adaptation to follow the shock that moves from the upper to the lower surface of the wing and also to refine the grid near the trailing edge, where vorticity is released because of the pitching motion. The Hessian of the pressure and of the magnitude of the vorticity are mixed (with relative weights of 0.83 and 0.17, respectively) and 3 steps are carried out while computing the refinement and coarsening thresholds by means of the multi-passage strategy.

As in the bi-dimensional test, two different non-dimensional time steps are used, $\Delta t = \Delta t_0$ and $\Delta t = 2\Delta t_0$, which result respectively in the 200 and 100 iterations for the whole simulation time $T^{\text{sim}} = 4\pi/\omega = 87.1778$.

Figure 6.8 displays the grids obtained at different time steps for the simulation performed with Δt_0. The figure displays the initial time $t = 0$ at the top-left corner, then starting from the top-right corner it shows the grid during the second period, more precisely at $t = 1.0\,T$, $t = 1.25\,T$, $t = 1.5\,T$, $t = 1.75\,T$ and $t = 2\,T$. With respect to the bi-dimensional case, to avoid an excessive increase in the number of grid points, the coarsening feature of Mmg3d is exploited. With the adaptation parameters $k_R = 2.92$ and $k_C = 0.7$, the number of grid points oscillates between approximately 100 000 and 190 000 during the whole simulation. The maximum is reached at $t = 1.25T$ and $t = 1.75T$ when the incidence is maximum (in absolute value) and the shock has the strongest intensity. Figure 6.9 shows the grid near the leading and trailing edge at $t = 1.25T$, $t = 1.75T$ and $t = 2T$, to highlight the deformation that the mesh experiences in order to follow the movements of the wing, which reaches the maximum amplitude at the trailing edge.

Figure 6.10 shows the contour lines of the Mach number at the time steps shown also in Figure 6.8. Thanks to mesh adaptation the shock over the upper and the lower airfoil surfaces is well captured. As in the bi-dimensional test, the slip line behind the trailing edge is not well resolved, however the refinement that occurs in this region indicates that the error estimator is capable of detecting the release of vorticity, but much more nodes are required to resolve the shear. Unfortunately, as for the M6 wing test, the computational time is large. The simulation with 200 time-steps takes $19d - 18h : 2m$ on 12 Cores of Intel Xeon CPU X5650, 2.67GHz, while the 100 time-steps simulations takes $12d{-}10h : 36m$. In all tests, at each time step, the solution of the System (2.33) is the most time-consuming part.

The bi-dimensional character of the flow is well reproduced in the three-dimensional results. As it can be noticed in the Mach contour plot of Figure 6.11, the solution along all the wing span is practically the same. Figure 6.12 and Figure 6.13 highlight better this result for several time steps. The former shows

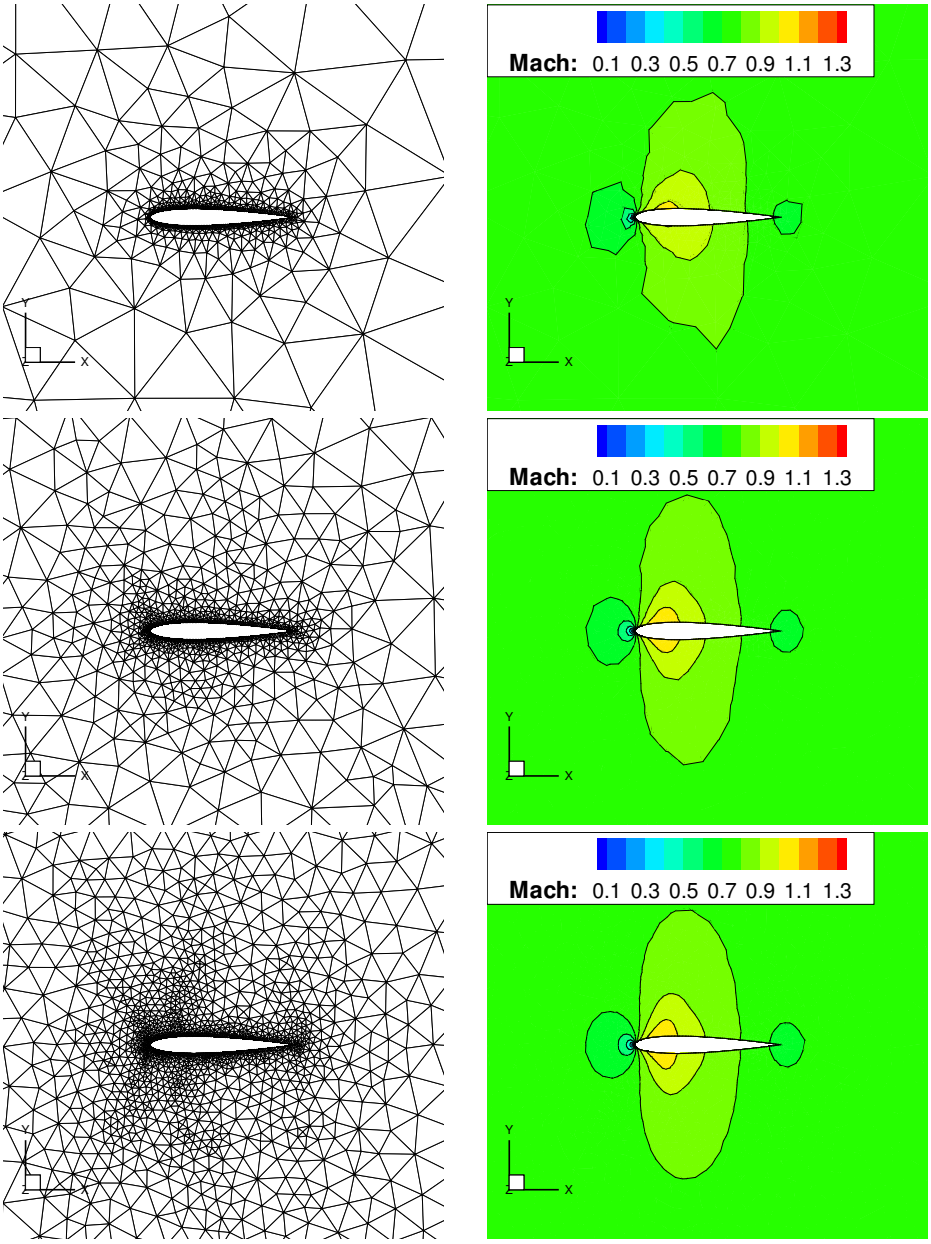

Figure 6.6: Grid and Mach contour plots of the transonic steady flow around the infinite-span NACA 0012 wing after different adaptation cycles, plane at $z = c$. In the first line, initial grid and solution; in the second and third line, grid and solution after 1 and 2 adaptation cycles.

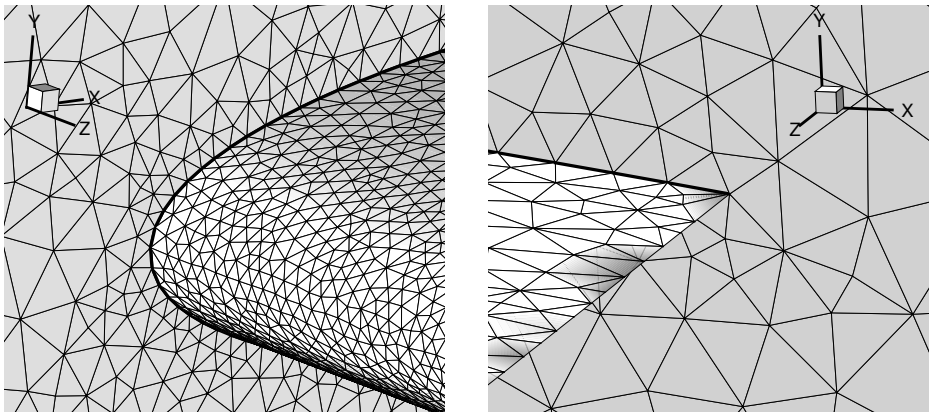

Figure 6.7: Details of the grid near the leading and trailing edge for the transonic steady flow around the infinite-span NACA 0012 after 2 adaptation cycle.

three pressure iso-lines at three different sections, namely at $z/b = 0.25$, at $z/b = 0.5$ and at $z/b = 0.75$, while the latter displays the pressure coefficients over the wing surface at the same three sections. A nearly null deviation can be detected between different wing sections.

The results obtained for different values of the time step Δt are compared in Figure 6.14, which depicts the lift coefficient versus the angle of attack, along with the results obtained in the bi-dimensional simulation of Subsection 5.1.2. No significant differences can be noticed between the curves. However, it should be noticed that, increasing the time step, the refinement parameters is increased to $k_R = 3.1$ to avoid an excessive growth of the number of grid points.

Furthermore, the effect of the prediction step is investigated. In addition to the simulation previously described in which a prediction step is performed, an additional simulation is carried out skipping the prediction phase. The results are compared through the curve C_L-α of Figure 6.14. Although only one solution computation is performed at each time step, instead of two as in the simulation with the prediction step, a comparable computational time was required. More precisely, the 200 time-steps simulation takes $20d - 1h : 11m$ on 12 Cores of Intel Xeon CPU X5650, 2.67GHz. This quite surprising result is due to the fact that with the prediction step the computation of the solution on the adapted grid is faster because the initial guess solution (computed during the prediction step) is close to the final one.

Moreover, the importance of the prediction step is confirmed by the impossibility to obtain the solution of the 100 times-steps simulations with the same parameters but without the prediction step. Probably, due to high unsteadiness of the problem, with the time step $\Delta t = 2\Delta t_0$ the adaptation is carried out on a mesh that is too different from the actual one and therefore the convergence is extremely difficult to achieve during the computation of the solution over the new adapted grid.

Finally, the conservativeness of the proposed adaptive strategy is assessed by evaluating the integral value of the density over the domain during the simulation, which is shown in Figure 6.15 for the three different tests, i.e. using the time steps Δt_0 and $2\Delta t_0$ and without performing the prediction step. The variation around

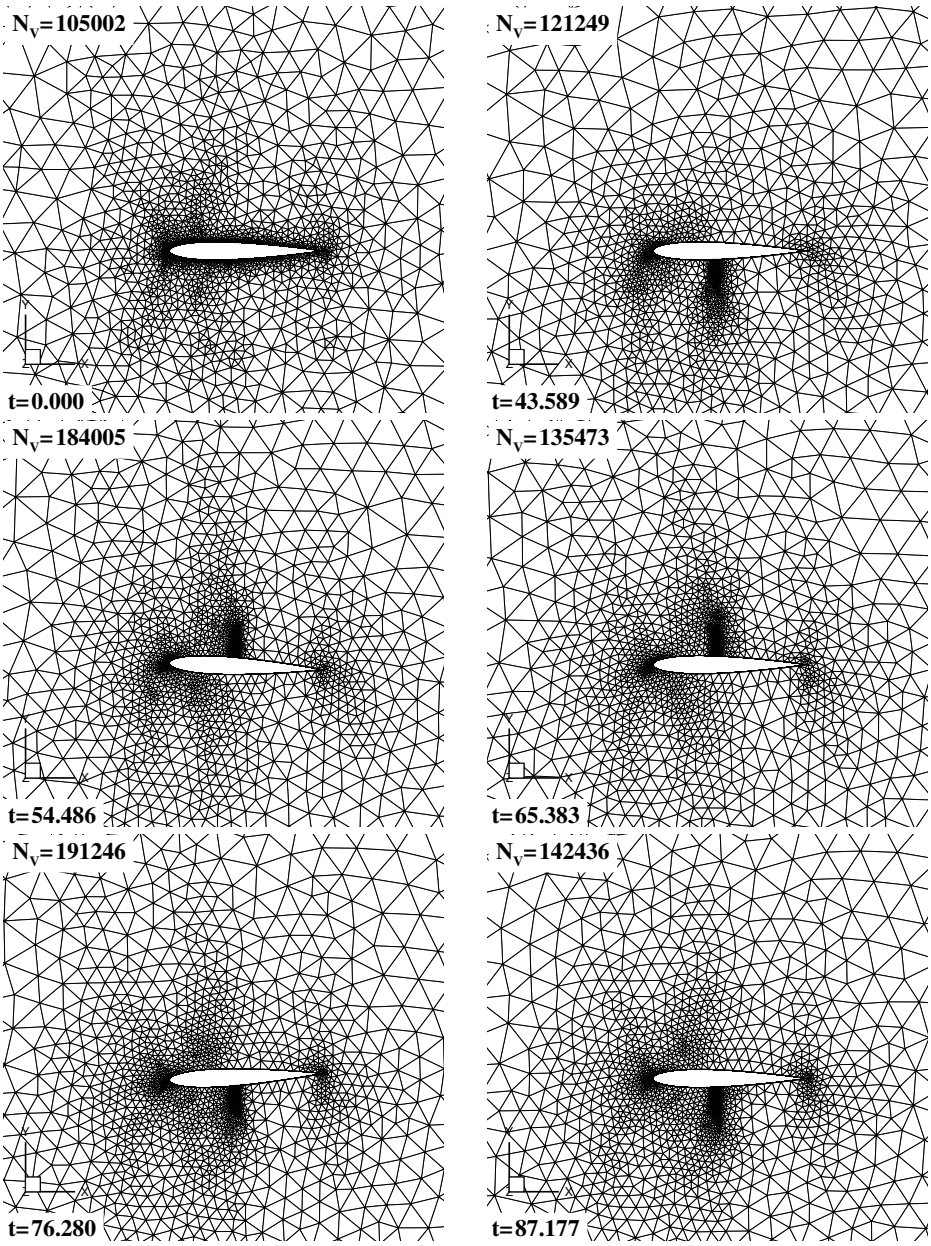

Figure 6.8: Detail of the computational grids for the pitching infinite-span NACA 0012 wing at different time steps using $\Delta t = \Delta t_0$, plane at $z = c$. From the top-left to the bottom-right: grid at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\mathrm{sim}}$.

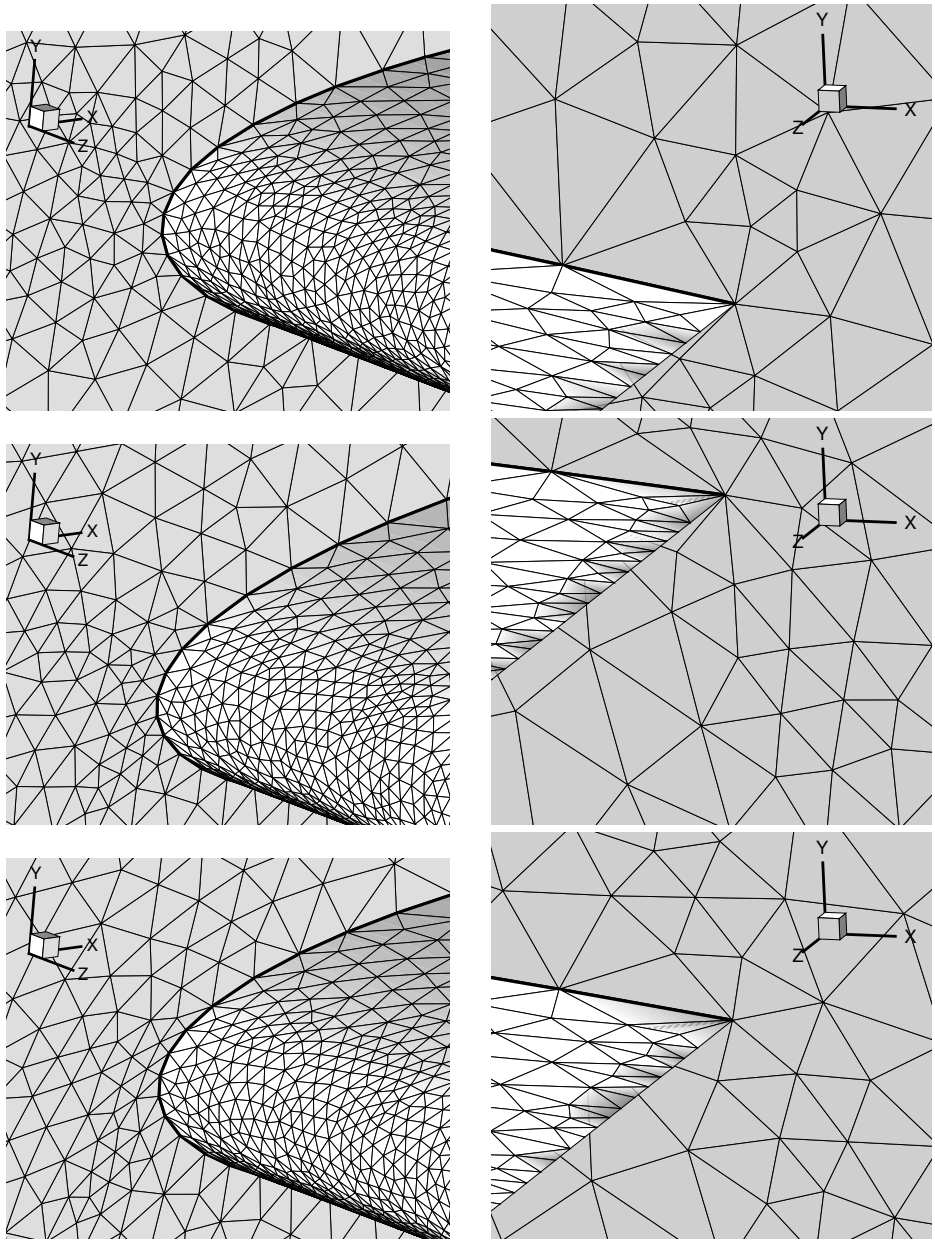

Figure 6.9: Details of the grids near the trailing and leading edge for the pitching infinite-span NACA 0012 wing at different time steps using $\Delta t = \Delta t_0$. From top to bottom: grid at $t = 1.25T$, $t = 1.75T$ and $t = 2T = T^{\mathrm{sim}}$.

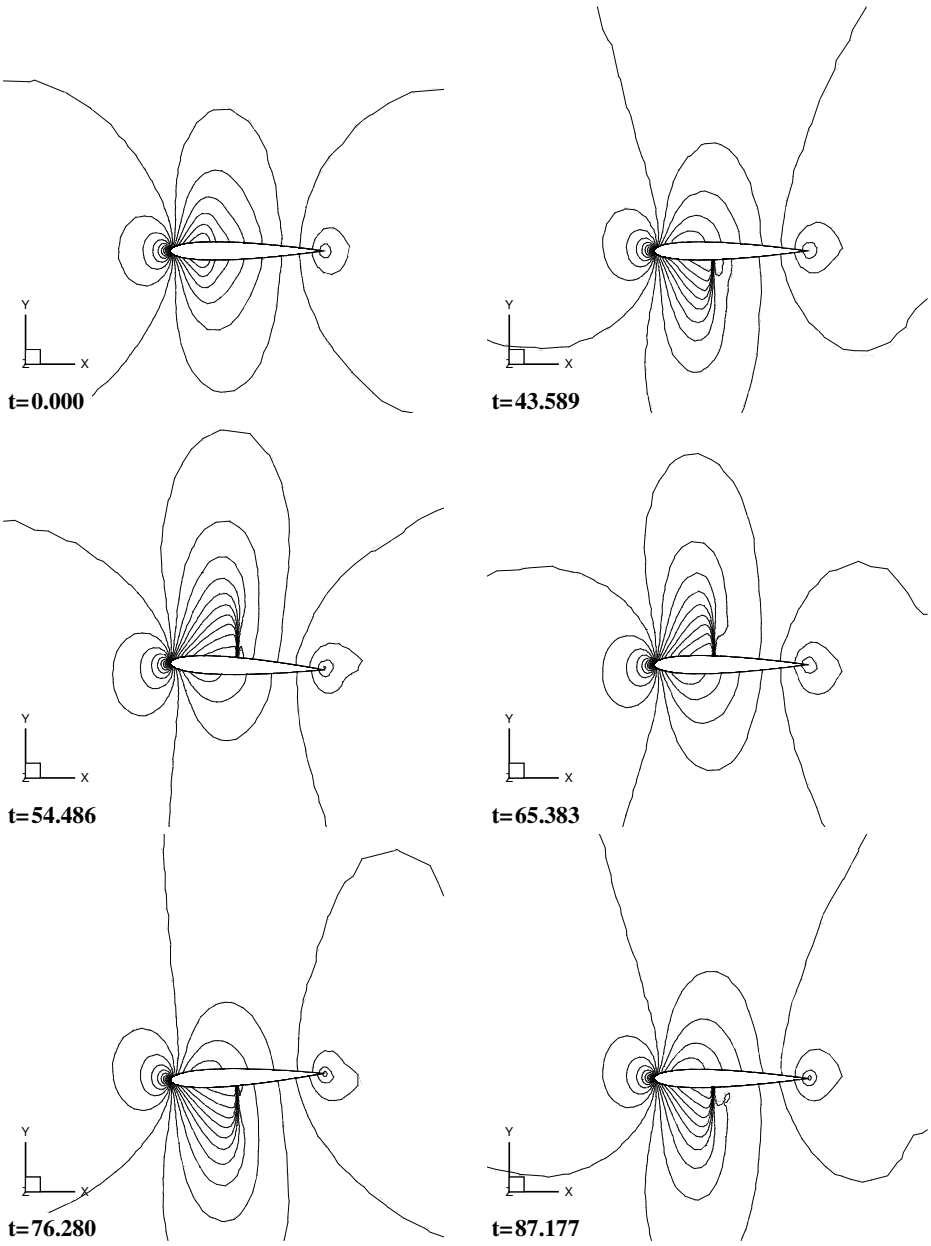

Figure 6.10: Mach contour lines of the flow around the pitching infinite-span NACA 0012 wing at different time steps using $\Delta t = \Delta t_0$, plane at $z = c$. From the top-left to the bottom-right: grid at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\mathrm{sim}}$.

the mean value is very limited, below the 0.05%, and the larger variation occurs in the initial part of the simulation. This fact is probably due to the different adaptation parameters used in the steady and unsteady simulations. After some time, this difference becomes irrelevant and the integral value remains constant, proving the conservative character of the adaptive scheme.

6.2.3 Unsteady simulation in the laboratory reference

The capability of dealing with large displacements of solid boundaries is now assessed. The same flow around the infinite-span NACA 0012 wing described in Subsection 6.2.1 is now simulated. Instead of performing the simulation in the wing reference frame as usual, the simulation is performed in the laboratory reference frame, namely the wing travels through the domain at the actual velocity and at the far-field quiescent air conditions are enforced.

The initial solution is obtained by subtracting to the solution obtained in the steady simulation the free-stream, both in term of the momentum $\boldsymbol{m}$ and of the total energy E^{t}. Then, a displacement of one chord in the negative x-direction, i.e. $\Delta x = -c$, is imposed to the wing with a velocity corresponding to $\mathrm{M}_\infty = 0.755$. The same angle of attack $\alpha_\infty = 0.016°$ is kept to be able to compare the results.

The whole simulation time, which corresponds to a non-dimensional time $T^{\mathrm{sim}} = 1.1294$, is divided into 125 time steps, so that the Courant number on the minimum allowed grid edge is 2. A Backward Euler scheme is used to perform time integration. The error estimator composed by the Hessian of the pressure and of the vorticity is used to drive solution-based adaptation, with $k_R = 2.91$ and $k_C = 0.7$. Two different simulations are carried out using different weights for the pressure and the vorticity in the compound error estimator: in the simulation labeled *Unsteady A* $w_P = 83\%$ and $w_\omega = 17\%$, while in the simulation labeled *Unsteady B* $w_P = 67\%$ and $w_\omega = 33\%$, where w_P and w_ω are the weights of the Hessian of the pressure and of the vorticity, respectively.

Figures 6.16 and 6.17 show the grid and the pressure contour lines at the beginning of the simulation *Unsteady A* and when one third, two thirds and the whole displacement is carried out. As expected the variation in the number of grid points is quite limited, since also the variations in the solution are small.

A quantitative comparison is displayed in Figure 6.18 which shows the variation of the lift coefficient during the unsteady simulations and the value obtained in the steady one. In both cases, the variations around the steady value remain limited, assessing the capability of the proposed approach to deal with large boundary displacement.

Finally, Figure 6.19 displays the variation in time of the integral value of the density over the domain, for both test cases. As for the pitching test case, the variation around the mean value is very limited, below the 0.05%, assessing the conservativeness of the proposed adaptive scheme.

6.3 Finite-span NACA 0012 wing

Finally, some simulations of the flow around the finite-span NACA 0012 wing are performed. Such a flow field is fully three-dimensional. Indeed, the difference between the pressure on the upper and lower wing surface results in a circulatory

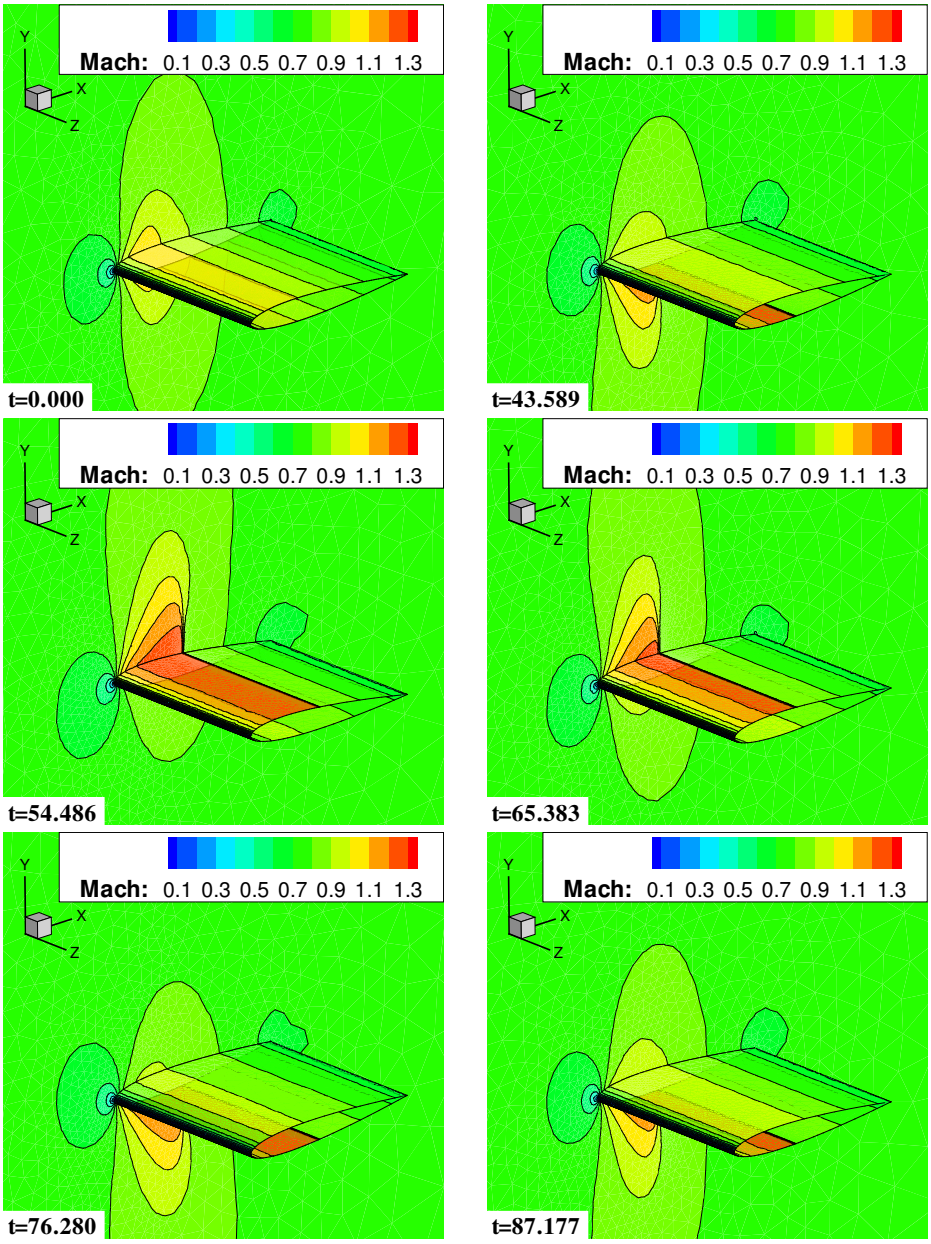

Figure 6.11: Mach contour plot of the flow around the pitching infinite-span NACA 0012 wing at different time steps using $\Delta t = \Delta t_0$. Only the wall at $z = 0$ and the upper wing surface are visible. From the top-left to the bottom-right: grid at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\mathrm{sim}}$.

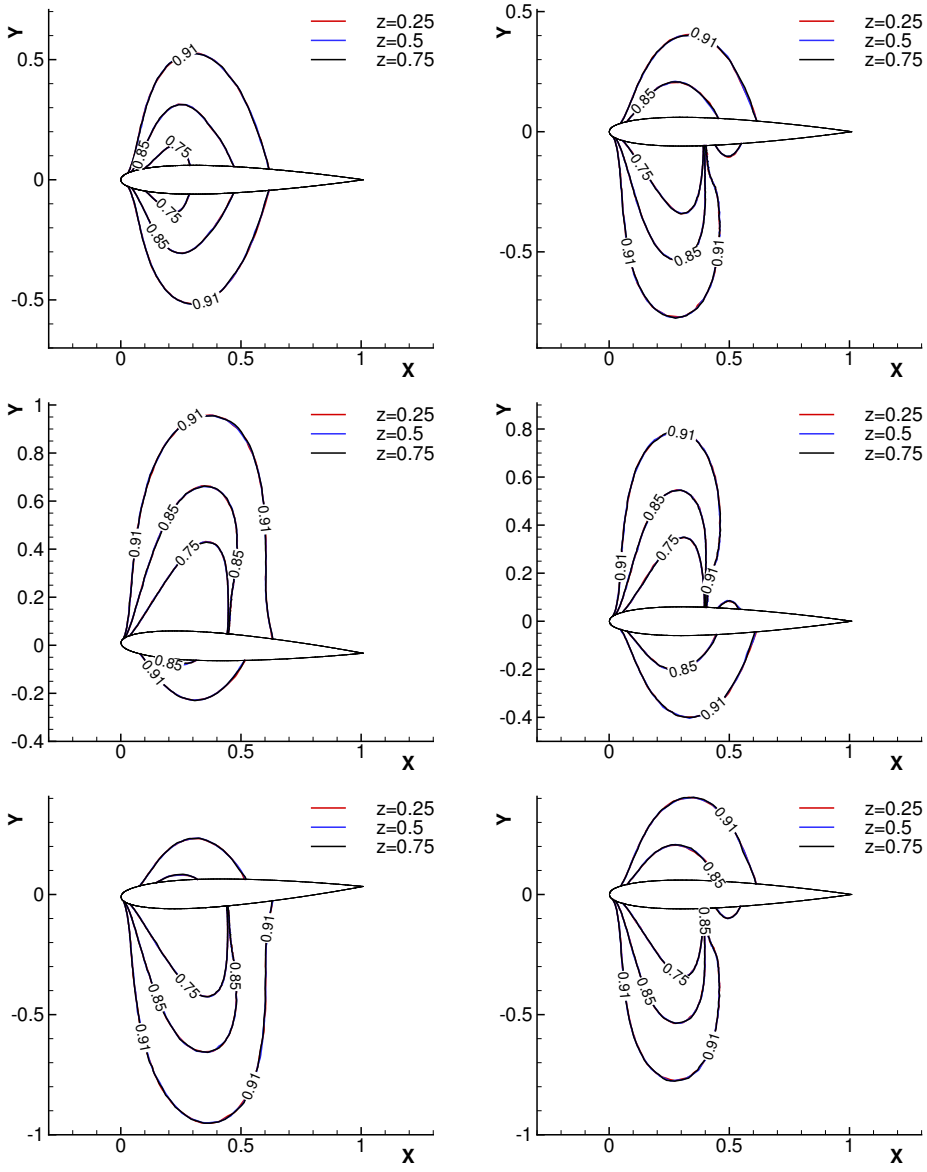

Figure 6.12: Pressure iso-lines of the flow around the pitching infinite-span NACA 0012 wing at different time steps. The iso-lines of the pressure reduced with respect to the free-stream value at $P/P_\infty = 0.75$, $P/P_\infty = 0.85$ and $P/P_\infty = 0.91$ at three different section are compared. From the top-left to the bottom-right: grid at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\text{sim}}$.

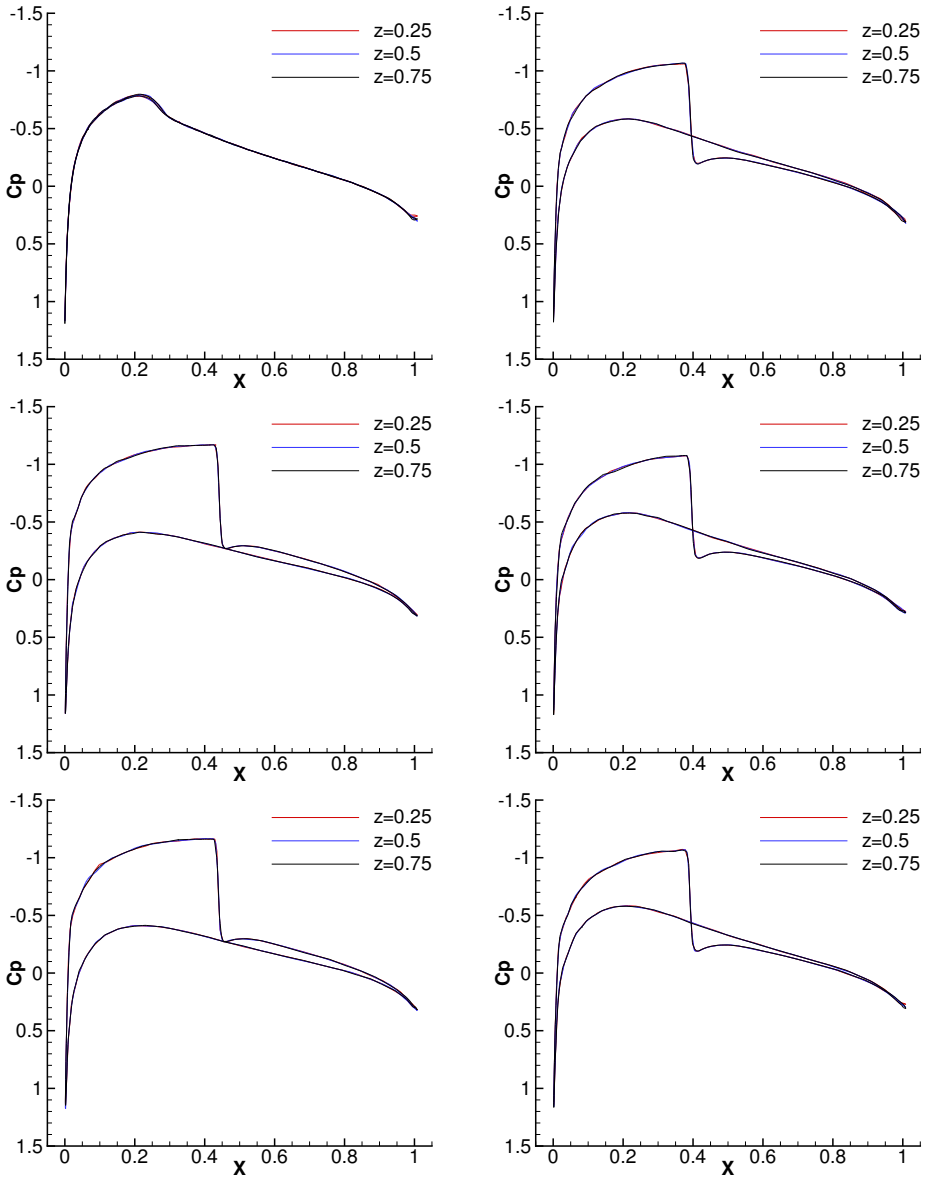

Figure 6.13: Pressure coefficient over the pitching infinite-span NACA 0012 wing at different time steps at three different section are compared. From the top-left to the bottom-right: grid at $t = 0T$, $t = 1.0T$, $t = 1.25T$, $t = 1.5T$, $t = 1.75T$ and $t = 2T = T^{\text{sim}}$.

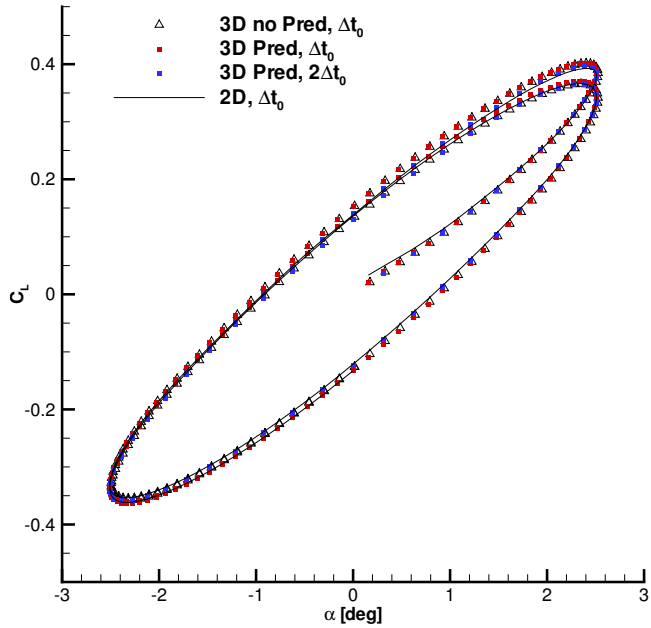

Figure 6.14: Lift coefficient hysteresis of the transonic pitching infinite-span NACA 0012 wing. The results obtained with the two different time-steps and without the prediction step (with $\Delta t = \Delta t_0$) are shown, along with the bi-dimensional results obtained in Subsection 5.1.2.

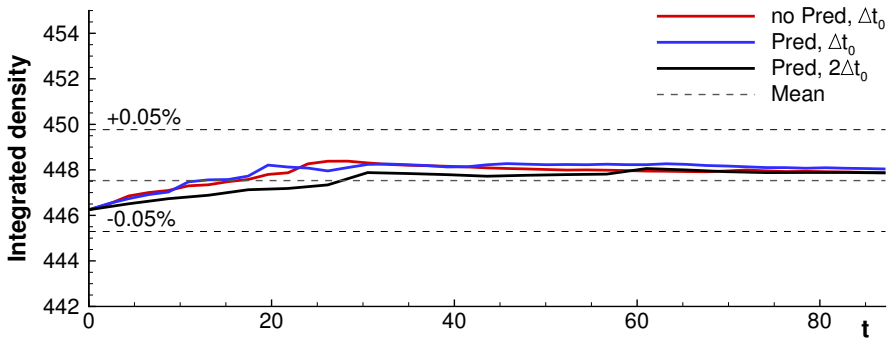

Figure 6.15: Variation in time of the integral value of the density over the domain for the transonic pitching infinite-span NACA 0012 wing. The results obtained with the two different time-steps and without the prediction step (with $\Delta t = \Delta t_0$) are shown. The mean value is also displayed, along with the variations $\pm 0.05\%$ around the mean value.

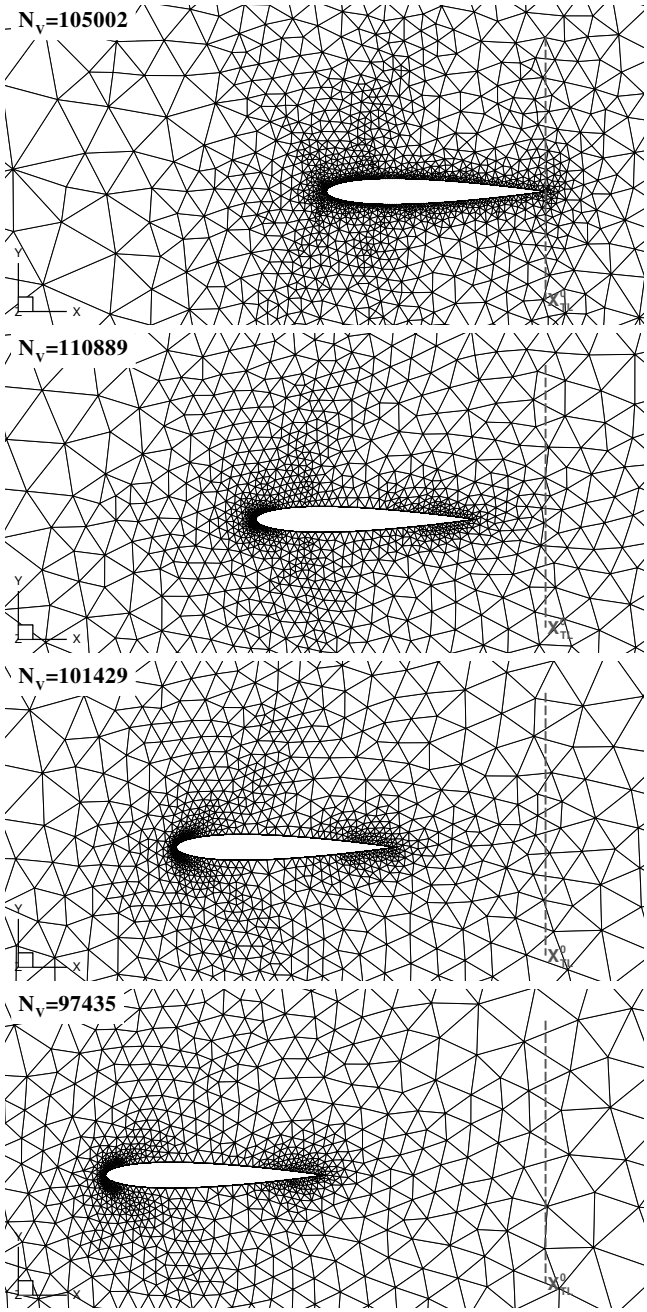

Figure 6.16: Mesh on the plane $z = c$ at the beginning, at 33%, at 66% and the end of one chord displacement for the unsteady simulation of the infinite-span NACA 0012 wing at $M_\infty = 0.755$ in the laboratory frame. The initial position of the trailing edge is indicated by the dashed vertical line.

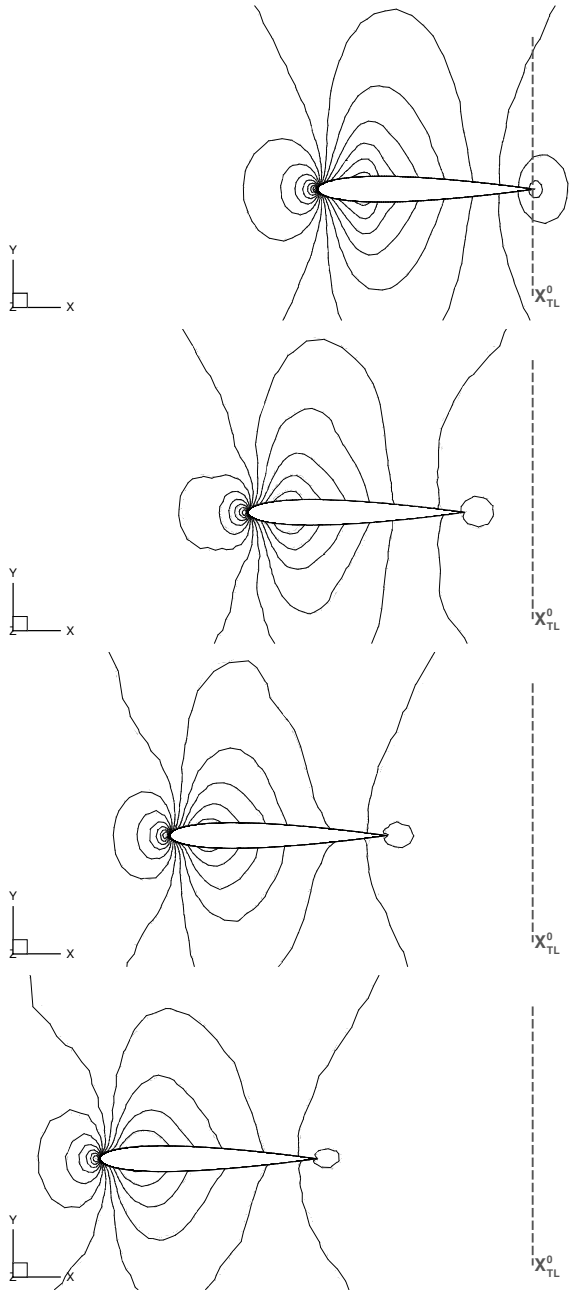

Figure 6.17: Pressure contour lines on the plane $z = c$ at the beginning, at 33%, at 66% and the end of one chord displacement for the unsteady simulation of the infinite-span NACA 0012 wing at $M_\infty = 0.755$ in the laboratory frame. The initial position of the trailing edge is indicated by the dashed vertical line.

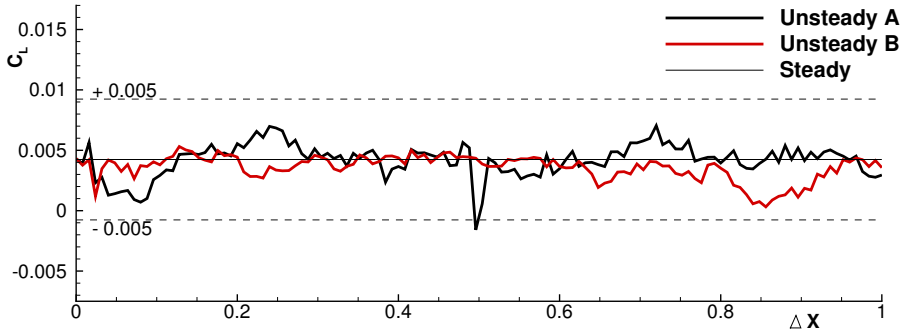

Figure 6.18: Lift coefficient for the simulation of the infinite-span NACA 0012 wing at $M_\infty = 0.755$ in the laboratory frame. The results of the unsteady simulation are compared with the value obtained in the steady simulation, $C_L = 0.0043$. Two different set of adaptation parameters are compared.

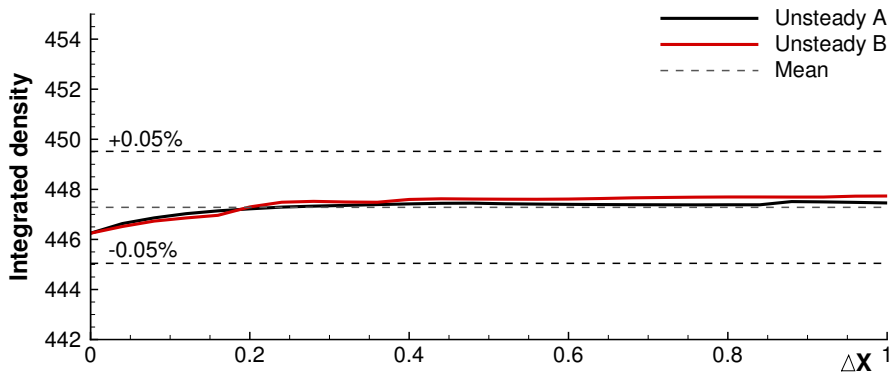

Figure 6.19: Variation in time of the integral value of the density over the domain for the unsteady simulation of the infinite-span NACA 0012 wing in the laboratory frame. The results obtained with the two different set of adaptation parameters are compared. The mean value is also displayed, along with the variations $\pm 0.05\%$ around the mean value.

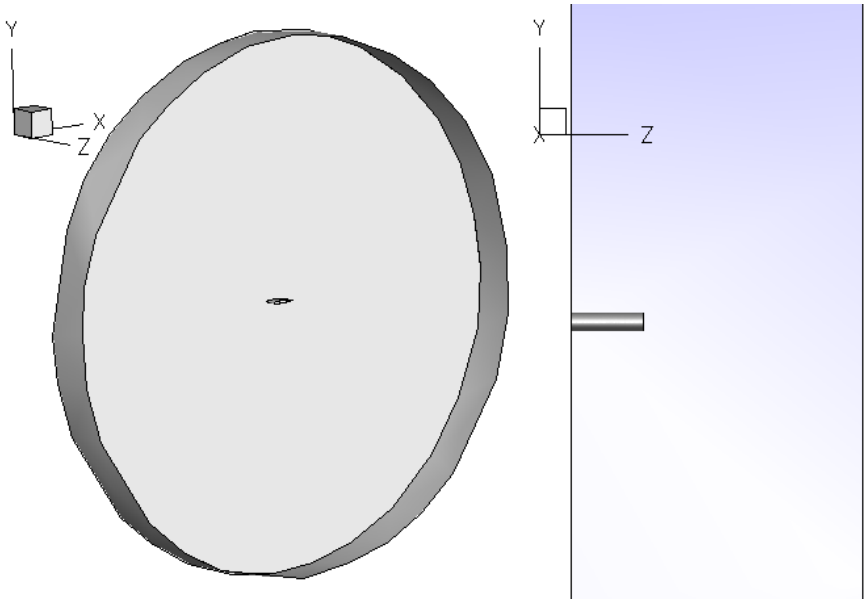

Figure 6.20: Computational domain for the simulations of the flow around the finite-span NACA 0012 wing.

motion near the wing tips, where vortices develop. The aerodynamic coefficients of a finite span wing are significantly different from the ones of the airfoils, mainly because of the induced drag due to the down-wash that occurs near the wing ends. Beside fixed-wing applications, this kind of simulations are of interest for the rotor-craft CFD community. For instance, a periodically pitching finite-span wing can represent the variation in the angle of attack that a rotor blade experiences during a complete rotation [85].

The numerical investigations presented in this section refer to a NACA 0012 wing with a span of half chord, namely $b = 0.5c$. The computational domain, represented in Figure 6.20, consists, in addition to the wing, in three surfaces: the wall at the wing root ($z = 0$), the opposite plane at $z = 2c$ and the cylindrical surface representing the far-field. As for the infinite-span wing tests, the polytropic ideal gas model and the high-order scheme are used.

6.3.1 Steady simulation

A steady simulation of the finite-span wing is first performed at Mach $M_\infty = 0.755$ and $\alpha = 0.016°$. The initial grid contains 15 484 nodes and 58 261 elements. Six mesh adaptation cycles are carried out by using an error estimator based on the Hessian of the pressure and of the vorticity. With respect to the infinite-span wing test, a more marked refinement is performed, especially near the trailing edge and the wing tip, where vortices are supposed to form.

Figure 6.21 displays the grid and the Mach contour plot at the beginning of the simulation and after three and six adaptation steps. As expected, the pressure distribution differs significantly from the one obtained in the infinite-span wing of Subsection 6.2.1, despite the equal free-stream conditions and the

TEST	k_R	k_C	w_P	w_ω	COMPUT. TIME
A	3.2	0.65	0.5	0.5	$3d-7h:54m$
B	2.9	0.68	0.77	0.23	$5d-9h:37m$
C	2.93	0.68	0.77	0.23	$4d-19h:48m$
D	2.9	0.65	0.77	0.23	$6d-12h:31m$
E	2.9	0.62	0.77	0.23	$7d-17h:16m$

Table 6.1: Combinations of adaptation parameters and strategies used in different test cases for the unsteady simulation of the transonic flow around the finite-span NACA 0012 wing. The error estimator is composed by mixed the Hessian of the pressure and of the vorticity according to the relative weights w_P and w_ω. The computational time required by the whole simulation is reported in the last column.

angle of attack. No shock is observed over the upper wing surface.

6.3.2 Unsteady simulation in the laboratory reference

The same transonic flow around the finite-span NACA 0012 wing of the previous subsection, i.e. at Mach $\mathrm{M}_\infty = 0.755$ and $\alpha = 0.016°$, is now simulated in the laboratory reference frame. As described in Subsection 6.2.3, a quiescent flow is enforced over the far-field boundary and the wing experiences a movement of one chord through the domain at the actual velocity. The time required by the whole movement is divided into 125 time steps and time integration is performed through the Backward Euler scheme.

Mesh adaptation is performed at each time step and different error estimators are tested. Figure 6.22 shows the grid at different times obtained using $k_R = 2.9$, $k_C = 0.65$ and three steps in the multi-passage strategy. During the simulations the number of grid points progressively decreases, because of the coarsening that occurs mainly in central part of the wing.

The capability of the conservative adaptive ALE scheme is confirmed in the present test by the limited variations in the pressure field and in the lift coefficient, respectively shown in Figures 6.23 and 6.24 at the beginning, at one third, at two thirds and at the end of the one chord displacement.

Different error estimators are compared in terms of lift coefficient. The results of the five combinations of adaptation parameters reported in Table 6.1 are compared in Figure 6.25. In all cases, the variations around the value obtained in the steady simulation are limited, proving that the validity of the conservative adaptive scheme is not dependent on a particular combination of adaptation parameters.

In such a test case, the capability of well resolving the grid where vorticity is released is an important feature. To this end, Figure 6.26 shows the iso-lines for $|\omega| = 0.7$, at different time steps. Unfortunately, the coarsening that occurs in the central part of the wing modifies improperly the grid, thus the capability of detecting vorticity deteriorates during the simulation. A better resolution may be achieved by a finer grid or a more intense refinement. However this possibility has not been investigated because the computational time, reported for the five simulations in Table 6.1, are quite large, at least at the current level of development. A more effective parallelization of the code is recommended to compute the solutions over a grid sufficiently fine to well resolve wing-tip vortices.

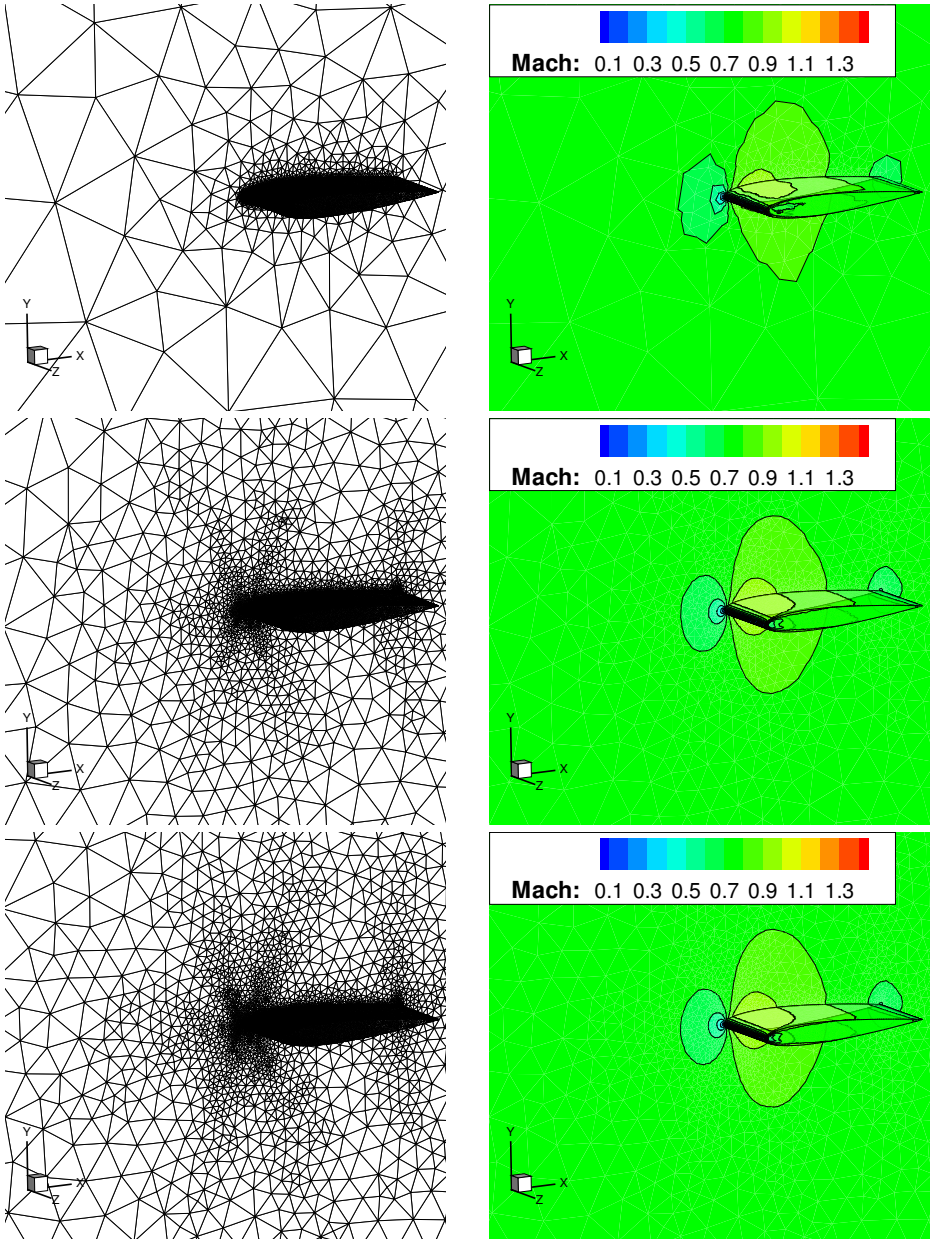

Figure 6.21: Grid and Mach contour plots of the transonic steady flow around the finite-span NACA 0012 wing after different adaptation cycles. In the first line, initial grid and solution; in the second and third line, grid and solution after 3 and 6 adaptation cycles.

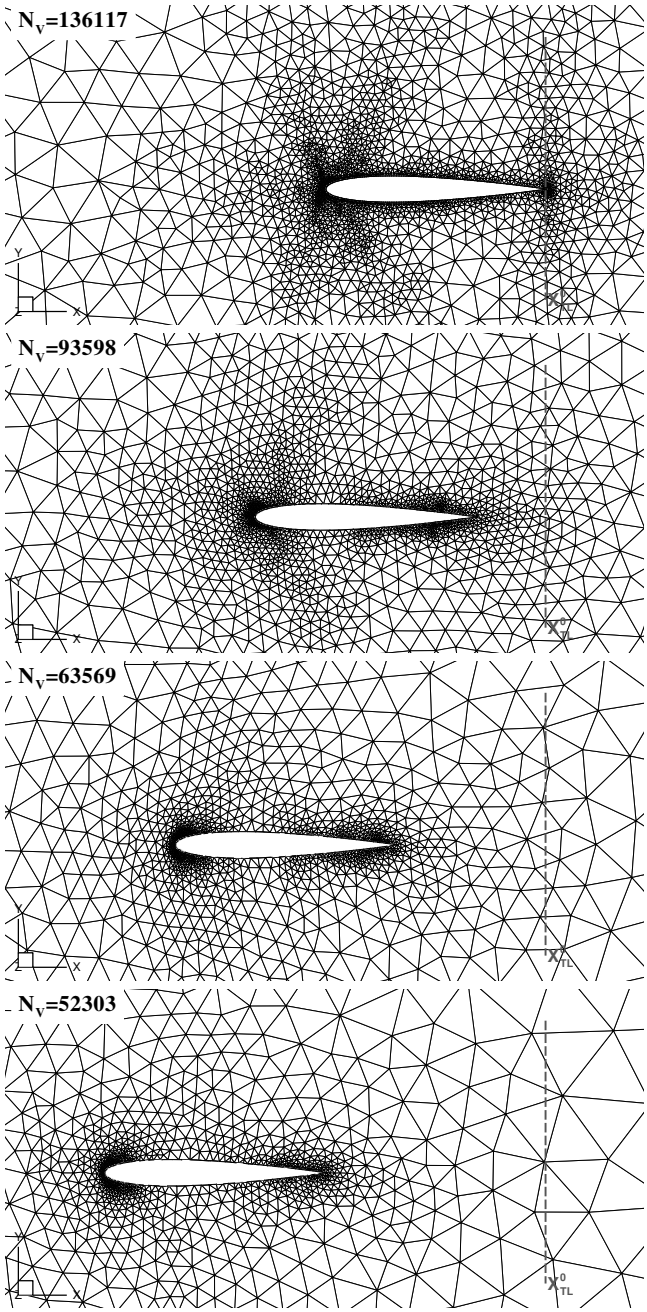

Figure 6.22: Surface mesh viewed from $z = 2c$ at the beginning, at 33%, at 66% and the end of one chord displacement for the unsteady simulation of the finite-span NACA 0012 wing at $M_\infty = 0.755$ in the laboratory frame. The initial position of the trailing edge is indicated by the dashed vertical line.

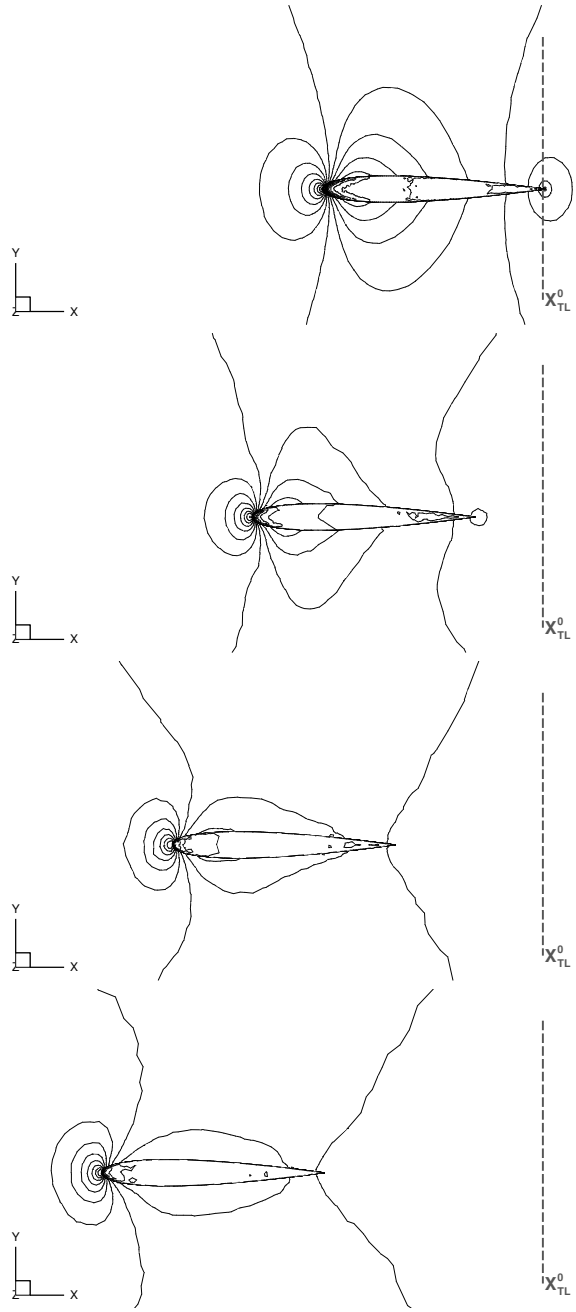

Figure 6.23: Pressure contour lines view from the plane $z = 2c$ at the beginning, at 33%, at 66% and the end of one chord displacement for the unsteady simulation of the finite-span NACA 0012 wing at $M_\infty = 0.755$ in the laboratory frame. The initial position of the trailing edge is indicated by the dashed vertical line.

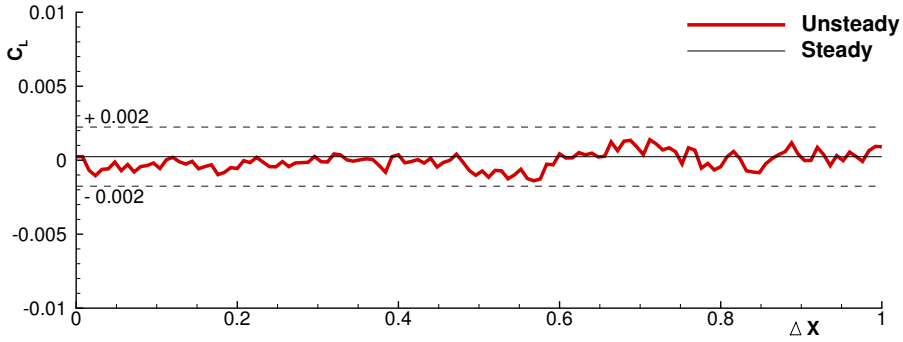

Figure 6.24: Lift coefficient for the simulation of the finite-span NACA 0012 wing at $M_\infty = 0.755$ in the laboratory frame. The results of the unsteady simulation are compared with the value obtained in the steady simulation, $C_L = 0.00024$.

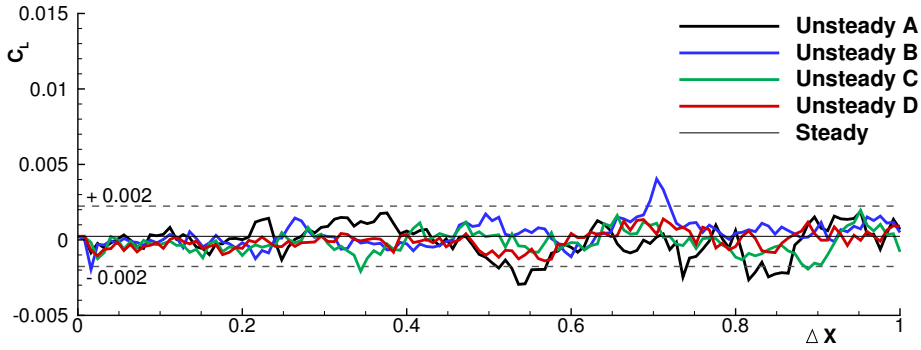

Figure 6.25: Lift coefficients for a finite-span NACA-0012 wing at $M_\infty = 0.755$. The different error estimators listed in Table 6.1 are compared.

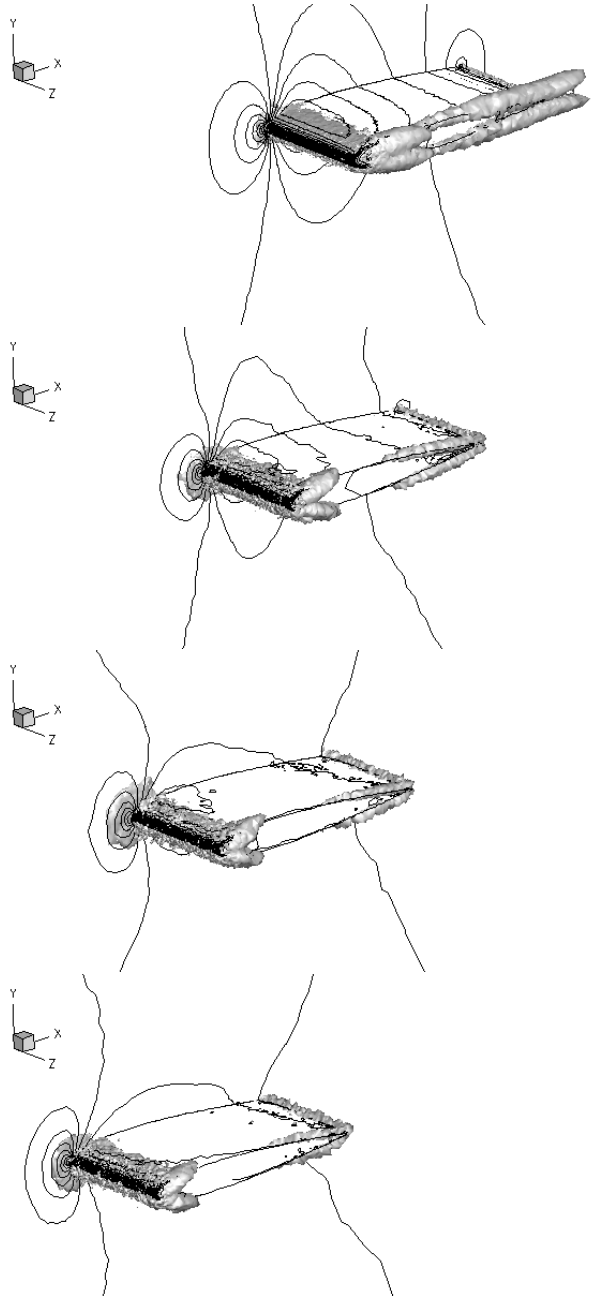

Figure 6.26: Iso-surface of vorticity magnitude $|\boldsymbol{\omega} \times \boldsymbol{v}| = 0.7$ at the beginning, at 33%, at 66% and the end of one chord displacement of the simulation of the finite-span NACA 0012 wing in the laboratory reference frame. Also pressure contour lines are shown.

Chapter 7

Conclusions and future developments

A novel interpolation-free conservative scheme for adaptive three-dimensional grids was presented for tetrahedral elements. The finite volume discretization of the Euler equations was obtained by a node-pair formulation, which allows to store all grid information in an edge-based data structure and to discharge element-related information. The Arbitrary Lagrangian-Eulerian formulation was exploited to enforce the governing equations over dynamic grids in a conservative fashion, even in presence of topology modifications due to mesh adaptation. Indeed, thanks to the three-steps procedure, proposed for bi-dimensional problems by Guardone and co-workers [59, 71] and here extended to three-dimensional problems, the volume changes due to local mesh adaptation are treated as a series of fictitious continuous deformations of the finite volumes that compose the computational domain. Thus, contrary to standard adaptive schemes, the proposed approach does not require any interpolations of the solution between different grids and the conservative and stability properties of the ALE scheme are preserved.

To overcome the numerical difficulties that arises when performing three-dimensional mesh adaptation, especially in terms of computational efficiency and mesh quality, the tetrahedral re-mesher Mmg3d developed by Dobrzynski [36] was exploited to perform node insertion, node deletion, edge swap and point displacement. The target grid spacing was computed on the basis of an error estimator obtained from the derivatives of some relevant flow variables, as for instance the Hessian of the pressure, of the Mach number or of the vorticity. The three-steps procedures requires that the volume change due to mesh adaptation is computed immediately after the modification, otherwise it may be impossible to retrieve the history of all modifications. Hence, the re-mesher was linked to the flow solver FlowMesh, actually under development at the Department of Aerospace Science and Technology of Politecnico di Milano, which was provided by a new data structure inter-operable with Mmg3d. Suitable communication functions were added to re-mesher to promptly pass the information about the performed local modification to the flow solver. Moreover, the grid velocity is computed as a function of the volume change so that the Discrete Geometric

Conservation Law is automatically fulfilled even if grid topology changes. To this end, for all possible local mesh adaptation operators a procedure that computes the volume change taking as inputs the positions of the involved nodes before and after the grid modification was developed and implemented in FlowMesh.

A robust deformation strategy to follow arbitrary large boundary displacements was developed starting from the elastic analogy proposed by Batina [12]. If the displacement performed with fixed connectivity is not sufficient, mesh adaptation is exploited to restore or increase the mesh quality. Moreover, when the boundaries of the computational domain experience large movements, a prediction step is carried out to compute the solution after mesh deformation, so that the target grid spacing refers to the updated geometry. Otherwise, the mesh is adapted with a significant delay with respect to the solution.

Different test cases were performed to evaluate the capabilities of the proposed strategy. First, the validity of the numerical procedure, in particular the exchange of information between the re-mesher and flow solver, was assessed in bi-dimensional problems within the ideal-gas region, since the communication functions implemented in the bi-dimensional version of the re-mesher were very similar to those implement in Mmg3d. Then, the three-dimensional adaptation strategy was assessed thanks to the piston-induced shock-tube flow problem, which presented two main difficulties: firstly the computational domain experienced a large deformation (at the end of the simulation the length of the domain had been reduced by more than an half), secondly the shock that formed at the beginning of the motion traveled through the domain. The numerical strategy successfully dealt with both tasks and a good agreement between the numerical results and the analytical solution, in particular in terms of shock location, was achieved with different combinations of adaptation parameters. Moreover, an investigation about the adaptation criteria in the so-called non-ideal compressible-fluid flow regime was carried out through different numerical simulations of supersonic under-expanded nozzle jets operating close to the liquid-vapor saturation curve. In this region, the fluid behavior departs significantly from the one predicted by the ideal gas model and the selection of the proper error estimator may be not straightforward because of the larger fluid compressibility, the non-ideal dependence of the speed of sound on the density and critical point phenomena. In the performed simulations, the estimators that included the derivatives of the Mach number or of the density proved to be more suitable to detect the relevant flow features.

Three-dimensional accurate simulations of aeronautical interest were carried out using the proposed conservative adaptive interpolation-free strategy. The steady transonic flow around the ONERA M6 wing were numerically computed to assess the capability of the present approach to accurately capture the relevant flow features. Thanks to six mesh adaptation cycles, the peculiar lambda-shock that formed in the considered flight conditions over the upper wing surface was well resolved, even if the initial grid spacing over the wing surface was uniform and quite coarse. Then, unsteady simulations of the flow around the pitching infinite-span NACA 0012 wing were performed. The bi-dimensional character of this flow was accurately reproduced in the three-dimensional results and a good agreement with the bi-dimensional results obtained for the pitching airfoil was obtained. To assess the capability of the proposed approach of dealing with large boundary displacements, the flow around the infinite-span NACA 0012

wing was numerically investigated from the laboratory reference frame, namely a quiescent flow was enforced over the boundary domain (i.e. the far-field) and the wing was moved through the domain at the flight velocity. With respect to the standard steady simulation in the wing reference frame, the results of the unsteady simulation showed a limited variation, more precisely the oscillations of the lift coefficient around the steady value are less than 0.005%. In both infinite-span wing problems, the conservativeness of the proposed method was proved by a limited variation of the integral value of the density over the domain. Finally, some simulations of the fully three-dimensional flow around the finite-span NACA 0012 wing were performed. This kind of simulations are of primary interest for the rotor-craft CFD community, since they can represent the periodic motion experienced by a rotor blade. The capability of accurately predicting the flow fields characterized by large boundary deformations was confirmed by the the unsteady simulation of the finite-span wing at Mach 0.755 in the laboratory reference frame.

All the performed tests proved that the developed numerical strategy is capable to successfully deal with arbitrarily large displacements of the boundary of the computational domain and to properly modify the grid spacing according to the solution in a conservative fashion. These features make the proposed approach well-suited to perform numerical investigations of geometrically complex and three-dimensional moving-body problems and of flow fields where the exact location of the relevant flow features is not known a priori, which are of primary importance in many applications, like for instance in rotor-craft simulations, turbo-machinery analysis, load separation problems and design of aeronautical control surfaces.

Unfortunately, no numerical investigations of complex geometries were carried out because the actual level of development of the flow solver is not sufficient to perform these simulations within an acceptable computational time. Indeed, only a rough parallelization of the solution computation procedure is available within the numerical tool. An optimization of the whole numerical strategy is required to thoroughly exploit the conservative three-dimensional adaptive scheme in complex domains. Besides a more efficient parallelization of the solution computation, also the mesh adaptation strategy has to be optimized to make a full use of the available computational resources. However, the distribution of the mesh adaptation procedure among several processors is not straightforward, because the re-mesher operators require neighborhood information. Therefore, an high amount of communications between adjacent sub-domains is required to achieve a mesh quality comparable to the one obtained in a sequential procedure. Nevertheless, sequential re-meshers can be used within a parallel framework, as proposed in [81], in an iteratively fashion. The initial mesh is partitioned into several sub-domains and an existing serial mesh adaptation algorithm is exploited on each sub-mesh. The interface made by the grid edges that cross different sub-domains is not modified by the sequential re-meshers. At the end of the iteration, the sub-domains are modified so that the nodes that were located on the interface are moved along with all the connected elements into a single sub-domain. Then, the whole algorithm starts again and mesh adaptation is carried out over the modified sub-domains. Indeed, at each iteration the interface moves to a new position and in some iterations mesh adaptation is performed over the whole domain.

A further development of the implemented strategy could be the enhancement of the error estimator. In the present thesis, error estimators based on the flow variables were chosen since they are simple and easy to be implemented. However, more accurate alternatives are available, like for instance the estimators based on the interpolation or discretization errors. To overcome the local character of these estimators, adjoint-based estimators could be implemented to obtain a local indicator for mesh adaptation related to the error in the output quantity of interest [46]. Moreover, anisotropic metric maps could be introduced to obtain a target grid spacing which provides different edge lengths in different directions. The implementation of anisotropic estimators would be quite straightforward since it affects a limited portion of the software and it would provide, at equal adaptation parameters, a saving in the number of the inserted nodes. Indeed, for instance, if a shock wave is detected, the grid would be refined only across the shock, but not along the other directions.

Furthermore, the present strategy can be easily extended to hybrid grids. Thanks to the grid transparency of the median-dual mesh approach and to the node-pair representation, in the flow solver all grid information is associated to the edges, so the use of different kinds of element would pose no problems in the solution computation. However, to efficiently perform mesh modifications the remesher Mmg3d works only with simplex elements, i.e. tetrahedra in 3D, which allow a fast research of adjacent elements. Nevertheless, the proposed approach can be extended to hybrid grids composed by a region of tetrahedral elements and other regions in which mesh adaptation is not required, like for instance the computational meshes used in viscous simulations composed by a region of prismatic elements with an high aspect ratio near the solid walls and tetrahedral elements elsewhere.

Bibliography

[1] R. Abgrall, H. Beaugendre, and C. Dobrzynski. An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques. *J. Comput. Phys.*, 257:83–101, 2014.

[2] M. Aftosmis. A second-order TVD method for the solution of the 3D Euler and Navier-Stokes equations on adaptively refined meshes. In *Thirteenth International Conference on Numerical Methods in Fluid Dynamics*, volume 414 of *Lecture Notes in Physics*, pp 235–239. Springer Berlin Heidelberg, 1993.

[3] M. Aftosmis and M. Berger. Multilevel Error Estimation and Adaptive h-Refinement for Cartesian Meshes with Embedded Boundaries. *AIAA paper*, 863:1–14, 2002.

[4] F. Alauzet. High-order methods and mesh adaptation for Euler equations. *Int. J. Numer. Meth. Fluids*, 56:1069–1076, 2008.

[5] F. Alauzet, P. Frey, P. George, and B. Mohammadi. 3D transient fixed point mesh adaptation for time-dependent problems: Application to CFD simulations. *J. Comput. Phys.*, 222:592–623, 2007.

[6] F. Alauzet, P. L. George, B. Mohammadi, P. Frey, and H. Borouchaki. Transient fixed point-based unstructured mesh adaptation. *Int. J. Numer. Meth. Fluids*, 43:729–745, 2003.

[7] M. Arora and P. L. Roe. On postshock oscillations due to shock capturing schemes in unsteady flows. *J. Comput. Phys.*, 130(1):25–40, 1997.

[8] I. Babuška and M. Suri. The p and h-p versions of the finite element method, basic principles and properties. *SIAM Rev.*, 36:578–632, 1994.

[9] T. J. Baker. Mesh Movement and Metamorphosis. *Eng. Comput.*, 18:188–198, 2002.

[10] T. J. Baker. Mesh adaptation strategies for problems in fluid dynamics. *Finite Elem. Anal. Des.*, 25(3–4):243 – 273, 1997.

[11] T. J. Barth and D. Jespersen. The design and application of upwind schemes on unstructured meshes. In *27th AIAA Aerospace Sciences Meetings*, pp 1–13, Reno, 1989. AIAA.

[12] J. T. Batina. Unsteady Euler airfoil solutions using unstructured dynamic meshes. *AIAA journal*, 28(8):1381–1388, 1990.

[13] T. Belytschko, W. K. Liu, and B. Moran. *Nonlinear Finite Element for Continua and Structures*. John Wiley & Sons, 2000.

[14] J. Benek, J. Steger, and F. Dougherty. A Flexible Grid Embedding Technique with Application to the Euler Equations. *AIAA paper*, 83-1944:373–382, 1983.

[15] M. J. Berger and A. Jameson. Automatic adaptive grid refinement for the Euler equations. *AIAA journal*, 23(4):561–568, 1985.

[16] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53(3):484–512, 1984.

[17] H. Borouchaki and P. Frey. Simplification of surface mesh using Hausdorff envelope. *Comput. Methods Appl. Mech. Engrg.*, 194:4864–4884, 2005.

[18] H. Borouchaki, P.-L. George, F. Hecht, P. Laug, and E. Saltel. Delaunay mesh generation governed by metric specifications. Part I. Algorithms. *Finite Elem. Anal. Des.*, 25:61–83, 1997.

[19] H. Borouchaki, F. Hecht, and P. Frey. Mesh gradation control. *Int. J. Numer. Meth. Eng.*, 43:1143–1165, 1998.

[20] H. B. Callen. *Thermodynamics and an introduction to thermostatistics.* Wiley, Second edition, 1985.

[21] G. Carpentieri. *An Adjoint-Based Shape-Optimization Method for Aerodynamic Design.* PhD thesis, Technische Universiteit Delft, Netherlands, 2009.

[22] M. Castro-Díaz, F. Hecht, B. Mohammadi, and O. Pironneau. Anisotropic unstructured mesh adaption for flow simulations. *Int. J. Numer. Meth. Fluids*, 25(4):475–491, 1997.

[23] S. Choi, J. Alonso, and E. van der. Weide. Numerical and mesh resolution requirements for accurate sonic boom prediction of complete aircraft configurations. *AIAA paper*, 1060:1–33, 2004.

[24] P. Colonna and T. P. der. Stelt. FluidProp: A program for the estimation of thermophysical properties of fluids. Energy Technology Section, Delft University of Technology, The Netherlands (www.fluidprop.com), 2005.

[25] P. Colonna and A. Guardone. Molecular Interpretation of nonclassical gasdynamics of dense vapors under the van der Waals model. *Phys. Fluids*, 18(5):56101–1–14, 2006.

[26] T. Coupez. Metric construction by length distribution tensor and edge based error for anisotropic adaptive meshing. *J. Comput. Phys.*, 230:2391–2405, 2011.

[27] P. I. Crumpton and M. B. Giles. Implicit time-accurate solutions on unstructured dynamic grids. *Int. J. Numer. Meth. Fluids*, 25(11):1285–1300, 1997.

[28] J. Dannenhoffer and J. Baron. Adaptation procedures for steady state solution of hyperbolic equations. *AIAA Paper*, 84-000, 1984.

[29] C. Dapogny, C. Dobrzynski, and P. Frey. Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems. *J. Comput. Phys.*, 262:358–378, 2014.

[30] F. Dassi, S. Perotto, and L. Formaggia. A priori anisotropic mesh adaptation on implicitly defined surfaces. Technical Report 51, MOX, Politecnico di Milano, Itay, 2004.

[31] A. de. Boer, M. van der. Schoot, and H. Bijl. Mesh deformation based on radial basis function interpolation. *Computers and Structures*, 85:784–795, 2007.

[32] S. Del Pino. Metric-based mesh adaptation for 2D Lagrangian compressible flows. *J. Comput. Phys.*, 230:1793–1821, 2011.

[33] C. Dobrzynski and P. Frey. Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations. In R. Garimella, editor, *Proceedings of the 17th International Meshing Roundtable*, pp 177–194. Springer Berlin Heidelberg, 2008.

[34] C. Dobrzynski. *Adaptation de Maillage anisotrope 3D et application à l'aérothermique des bâtiments.* PhD thesis, Université Pierre et Marie Curie, Paris VI, 2005.

[35] C. Dobrzynski, C. Dapogny, P. Frey, and A. Froehly. *Mmg3d [computer software]*.

[36] C. Dobrzynski and P. Frey. Anisotropic Delaunay mesh adaptation for unsteady simulations. In *Proceedings of the 17th international Meshing Roundtable*, pp 177–194. Springer, 2008.

[37] V. Dolejší. Anisotropic mesh adaptation for finite volume and finite element methods on triangular meshes. *Computing and Visualization in Science*, 1(3):165–178, 1998.

[38] V. Dolejší. Anisotropic hp-adaptive method based on interpolation error estimates in the H1-seminorm. *Applications of Mathematic*, 60(6):597–616, 2015.

[39] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodriguez-Ferran. Encyclopedia of Computational Mechanics Vol. 1: Fundamentals., Chapter 14: Arbitrary Lagrangian-Eulerian Methods, 2004.

[40] A. Duchini. Aspetti algoritmici della parallelizzazione di un codice fluidodinamico in formulazione nodepair per griglie mobili su architetture a memoria distribuita. Master's thesis, Politecnico di Milano, 2010-2011.

[41] S. Étienne, A. Garon, and D. Pelletier. Perspective on the geometric conservation law and finite element methods for ALE simulations of incompressible flow. *Journal of Computational Physics*, 228(7):2313 – 2333, 2009.

[42] C. Farhat, M. Lesoinne, and P. LeTallec. Load and motion transfer algorithms for fluid/ structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity. *Comput. Methods Appl. Mech. Engrg.*, 157:95–114, 1998.

[43] C. Farhat, C. Degand, B. Koobus, and M. Lesoinne. Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Comput. Methods Appl. Mech. Engrg.*, 163(1):231–245, 1998.

[44] C. Farhat, P. Geuzaine, and C. Grandmont. The Discrete Geometric Conservation Law and the Nonlinear Stability of ALE Schemes for the Solution of Flow Problems on Moving Grids. *J. Comput. Phys.*, 174:669–694, 2001.

[45] C. Farhat, M. Lesoinne, and N. Maman. Mixed explicit/implicit time integration of coupled aeroelastic problems: Three-field formulation, geometric conservation and distributed solution. *Int. J. Numer. Meth. Fluids*, 21(10):807–835, 1995.

[46] K. J. Fidkowski and D. L. Darmofal. Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics. *AIAA Journal*, 49(4):673–694, 2011.

[47] G. Fillola, M. L. Pape, and M. Montagnac. Numerical simulations around wing control surfaces. In *Proceedings of 24th International Congress of the Aeronautical Sciences (ICAS)*, Yokohama, Japan, 2004.

[48] L. Formaggia and F. Nobile. Stability analysis of second-order time accurate schemes for ALE–FEM. *Comput. Methods Appl. Mech. Engrg.*, 193(39–41):4097 – 4116, 2004.

[49] M. Fossati, A. Guardone, and L. Vigevano. A Node-Pair Finite Element/Volume Mesh Adaptation Technique for Compressible Flows. *Int. J. Numer. Meth. Fluids*, 70(8):1004–1026, 2012.

[50] B. Francois, M. Costes, and G. Dufour. Comparison of chimera and sliding mesh techniques for unsteady simulations of counter rotating open-rotors. In *20th International Society for Airbreathing Engines Conference (ISABE)*, Gothenburg, Sweden,, September 12-16 2011.

[51] L. A. Freitag and C. Ollivier-Gooch. Tetrahedral mesh improvement using swapping and smoothing. *Int. J. Numer. Meth. Eng.*, 40:3979–4002, 1997.

[52] P. Frey and P.-L. George. *Mesh Generation: Application to Finite Element*, volume 32. Wiley, 2010.

[53] P. Frey and F. Alauzet. Anisotropic mesh adaptation for CFD computations. *Comput. Methods Appl. Mech. Engrg.*, 194:5068–5082, 2005.

[54] M. Gadala, M. Movahhedy, and J. Wang. On the mesh motion for ALE modeling of metal forming processes. *Finite Elem. Anal. Des.*, 38(5):435 – 459, 2002.

[55] E. H. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin methods on hp-anisotropic meshes II: a posteriori error analysis and adaptivity. *Appl. Numer. Math.*, 59(9):2179 – 2194, 2009.

[56] P. Geuzaine, C. Grandmont, and C. H. Farhat. Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations. *J. Comput. Phys.*, 191(1):206–227., 2003.

[57] M. Giles. Stability analysis of a Galerkin/Runge–Kutta Navier–Stokes discretisation on unstructured tetrahedral grids. *J. Comput. Phys.*, 132:201–214, 1997.

[58] S. Giuliani. An algorithm for continuous rezoning of the hydrodynamic grid in Arbitrary Lagrangian-Eulerian computer codes. *Nucl. Eng. Des.*, 72(2):205 – 212, 1982.

[59] A. Guardone, D. Isola, and G. Quaranta. Arbitrary Lagrangian Eulerian formulation for two-dimensional flows using dynamic meshes with edge swapping. *J. Comput. Phys.*, 230(20):7706–7722, 2011.

[60] A. Guardone, M. Parsani, and L. Vigevano. Numerical simulations of underexpanded nozzle flows of dense gases. In *XVIII Congress of the Italian Association of Theoretical and Applied Mechanics (AIMETA)*, Brescia, Italy, 11-14 sep. 2007.

[61] A. Guardone and L. Quartapelle. Unstructured finite-volume high-resolution methods for conservation laws. Technical report, Department of aerospace science and technology - Politecnico di Milano, 2000.

[62] A. Guardone and L. Vigevano. Roe Linearization for the van der Waals Gas. *J. Comput. Phys.*, 175:50–78, 2002.

[63] H. Guillard and C. Farhat. On the significance of the geometric conservation law for flow computations on moving meshes. *Comput. Methods Appl. Mech. Engrg.*, 190:1467–1482, 2000.

[64] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M.-G. Vallet. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles. *Int. J. Numer. Meth. Fluids*, 32:725–744, 2000.

[65] R. Haber, M. S. Shephard, J. F. Abel, R. H. Gallagher, and D. P. Greenberg. A general two-dimensional, graphical finite element preprocessor utilizing discrete transfinite mappings. *Int. J. Numer. Meth. Eng.*, 17(7):1015–1044, 1981.

[66] A. Harten. High Resolution Schemes for Hyperbolic Conservation Laws. *J. Comput. Phys.*, 135:260–278, 1997.

[67] A. Haselbacher, J. J. McGuirk, and G. J. Page. Finite Volume Discretization Aspects for Viscous Flows on Mixed Unstructured Grids. *AIAA Journal*, 37(2):177–184, 1999.

[68] O. Hassan, K. Morgan, and N. Weatherill. Unstructured mesh methods for the solution of the unsteady compressible flow equations with moving boundary components. *Phil. Trans. R. Soc. A*, 365(1859):2531–2552, 2007.

[69] B. T. Helenbrook. Mesh deformation using the biharmonic operator. *Int. J. Numer. Meth. Eng.*, 56(7):1007–1021, 2003.

[70] D. Isola. *An Interpolation-Free Two-Dimensional Conservative ALE Scheme over Adaptive Unstructured Grids for Rotorcraft Aerodynamics*. PhD thesis, Politecnico di Milano, Department of Aerospace Engineering, 2012.

[71] D. Isola, A. Guardone, and G. Quaranta. Finite-volume solution of two-dimensional compressible flows over dynamic adaptive grids. *J. Comput. Phys.*, 285:1–23, 2015.

[72] A. Jameson. Time Dependent Calculations Using Multigrid, with Applications to Unsteady Flows Past Airfoils and Wings. *AIAA paper*, 1596:1–13, 1991.

[73] Y. Kallinderis and P. Vijayan. Adaptive refinement-coarsening scheme for three-dimensional unstructured meshes. *AIAA journal*, 31(8):1440–1447, 1993.

[74] Y. Kallinderis and J. Baron. Adaptation methods for a new Navier-Stokes algorithm. *AIAA journal*, 27(1):37–43, 1989.

[75] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1999.

[76] H. Katanoda, T. Handa, Y. Miyazato, M. Masuda, and K. Matsuo. Effect of Reynolds Number on Pitot-Pressure Distributions in Underexpanded Supersonic Freejets. *J. Propul. Power*, 17(4):940–942, July 2001.

[77] H. Katanoda, Y. Miyazato, M. Masuda, and K. Matsuo. Pitot pressures of correctly-expanded and underexpanded free jets from axisymmetric supersonic nozzles. *Shock Waves*, 10(2):95–101, 2000.

[78] B. Koren. Defect correction and multigrid for an efficient and accurate computation of airfoil flows. *J. Comput. Phys.*, 77:183–206, 1988.

[79] M. Kucharik, M. Shashkov, and B. Wendroff. An efficient linearity-and-bound-preserving remapping method. *J. Comput. Phys.*, 188(2):462–471, 2003.

[80] A. Kuzmin. On the lambda-shock formation on ONERA M6 wing. *International Journal of Applied Engineering Research*, 9(20):7029–7038, January 2014.

[81] C. Lachat, C. Dobrzynski, and F. Pellegrini. Parallel mesh adaptation using parallel graph partitioning. In *5th European Conference on Computational Mechanics (ECCM V)*, volume 3, pp 2612–2623. CIMNE-International Center for Numerical Methods in Engineering, 2014.

[82] J. Lambert. *Numerical Methods for Ordinary Differential Systems: the initial value problem*. John Wiley & Sons, Chichester, UK, 1991.

[83] R. Landon. NACA 0012 Oscillatory and Transient Pitching, Compendium of Unsteady Aerodynamic Measurements. Technical Report R-702, AGARD Report, 1982.

[84] C. L. Lawson. Transforming triangulations. *Discrete mathematics*, 3(4):365–372, 1972.

[85] J. Leishman. *Principles of Helicopter Aerodynamics*. Cambridge University Press, 2nd edition, 2006.

[86] M. Lesoinne and C. Farhat. Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations. *Comput. Methods Appl. Mech. Engrg.*, 134:71–90, 1996.

[87] R. J. LeVeque. *Numerical methods for conservation laws*. Birkhäuser, Basel, 1992.

[88] R. J. LeVeque. *Finite volume methods for conservation laws and hyperbolic systems*. Cambridge University Press, 2002.

[89] X. Li, M. S. Shephard, and M. W. Beall. 3D anisotropic mesh adaptation by mesh modification. *Comput. Methods Appl. Mech. Engrg.*, 194(48–49):4915 – 4950, 2005.

[90] P. T. Lin, T. J. Baker, L. Martinelli, and A. Jameson. Two-dimensional implicit time-dependent calculations on adaptive unstructured meshes with time evolving boundaries. *Int. J. Numer. Meth. Fluids*, 50:199–218, 2006.

[91] W. K. Liu, H. Chang, J.-S. Chen, and T. Belytschko. Arbitrary lagrangian-eulerian petrov-galerkin finite elements for nonlinear continua. *Comput. Methods Appl. Mech. Engrg.*, 68(3):259 – 310, 1988.

[92] R. Löhner. Mesh Adaptation in fluid Mechanics. *Eng. Fract. Mech.*, 50(5/6):819–847, 1995.

[93] R. Löhner and C. Yang. Improved ALE mesh velocities for moving bodies. *Commun. Numer. Meth. En.*, 12(10):599–608, October 1996.

[94] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, and S. Galera. ReALE: A reconnection-based arbitrary-Lagrangian–Eulerian method. *J. Comput. Phys.*, 229:4724–4761, 2010.

[95] L. Margolin and M. Shashkov. Second-order sign-preserving conservative interpolation (remapping) on general grids. *J. Comput. Phys.*, 184(1):266 – 298, 2003.

[96] D. Mavriplis. On convergence acceleration techniques for unstructured meshes. *AIAA paper*, 1998.

[97] D. J. Mavriplis. Unstructured grid techniques. *Annu. Rev. Fluid Mech.*, 29(1):473–514, 1997.

[98] D. J. Mavriplis and Z. Yang. Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes. *J. Comput. Phys.*, 213(2):557–573, 2006.

[99] D. Mavriplis. Adaptive meshing techniques for viscous flow calculations on mixed element unstructured meshes. *Int. J. Numer. Meth. Fluids*, 34(2):93–111, 2000.

[100] W. McCroskey. Unsteady Airfoils. *Annu. Rev. Fluid Mech.*, 14:285–311, 1982.

[101] R. Mittal and G. Iaccarino. Immersed Boundary Methods. *Annu. Rev. Fluid Mech.*, 37:239–261, 2005.

[102] D. Muffo, G. Quaranta, A. Guardone, and P. Mantegazza. Interface velocity consistency in time-accurate flow simulations on dynamic meshes. Technical Report Scientific Report DIA-SR 07-01, Politecnico di Milano, Italy, 2007.

[103] G. Ni, S. Jiang, and K. Xu. Remapping-free ALE-type kinetic method for flow computations. *J. Comput. Phys.*, 228:3154–3171, 2009.

[104] B. Nkonga. On the conservative and accurate CFD approximations for moving meshes and moving boundaries. *Comput. Methods Appl. Mech. Engrg.*, 190:1801–1825, 2000.

[105] C. Pain, A. Umpleby, C. de. Oliveira, and A. Goddard. Tetrahedral mesh optimisation and adaptivity for steady-state and transient finite element calculations. *Comput. Methods Appl. Mech. Engrg.*, 190(29–30):3771 – 3796, 2001.

[106] M. Pelanti, L. Quartapelle, and L. Vigevano. A review of entropy fixes as applied to Roe's linearization. Technical report, Department of aerospace science and technology - Politecnio di Milano, 2001.

[107] J. Peraire, J. Peiró, and K. Morgan. Adaptive Remeshing for Three-Dimensional Compressible Flow Computations. *J. Comput. Phys.*, 103:262–285, 1992.

[108] J. Peraire, M. Vahdati, K. Morgan, and O. Zienkiewicz. Adaptive remeshing for compressible flow computations. *J. Comput. Phys.*, 72(2):449 – 466, 1987.

[109] G. Quaranta, P. Masarati, and P. Mantegazza. A conservative mesh-free apporach for fluid-structure interface problems. In *Int. Conf. on Computational Methods for Coupled Problems in Science and Engineering*, Barcelona, 2005.

[110] M. M. Rai. A conservative treatment of zonal boundaries for Euler equation calculations. *J. Comput. Phys.*, 62(2):472–503, 1986.

[111] R. D. Rausch, J. T. Batina, and H. T. Yang. Spatial adaptation of unstructured meshes for unsteady aerodynamic flow computations. *AIAA Journal*, 30(5):1–15, May 1992.

[112] P. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *J. Comput. Phys.*, 135(2):250–258, 1997.

[113] A. Rohde. Eigenvalues and eigenvectors of the Euler equations in general geometries. *AIAA paper*, 2609:1–6, 2001.

[114] V. Schmitt and F. Charpin. Pressure distributions on the ONERA-M6-Wing at transonic Mach numbers. *Experimental data base for computer program assessment*, 4, 1979.

[115] V. Selmin. The node-centred finite volume approach: bridge between finite differences and finite elements,. *Comput. Methods Appl. Mech. Engrg.*, 102:107–138, 1993.

[116] V. Selmin and L. Formaggia. Unified construction of finite element and finite volume discretisations for compressible flows. *Int. J. Numer. Meth. Eng.*, 39:1–32, 1996.

[117] H. Tang, C. S. Jones, and F. Sotiropoulos. An overset-grid method for 3D unsteady incompressible flows. *J. Comput. Phys.*, 191:567–600, 2003.

[118] I. Thomas and T. Sonar. On a Second Order Residual Estimator for Numerical Schemes for Nonlinear Hyperbolic Conservation Laws. *J. Comput. Phys.*, 171:227–242, 2001.

[119] P. D. Thomas and C. K. Lombard. Geometric Conservation Law and Its Application to Flow Computations on Moving Grids. *AIAA Journal*, 17(10):1030–1037, October 1979.

[120] J. F. Thompson, B. K. Soni, and N. P. Weatherill. *Handbook of Grid Generation*. CRC Press, 1998.

[121] P. A. Thompson. A Fundamental Derivative in Gasdynamics. *Phys. Fluids*, 14(9):1843–1849, 1971.

[122] P. A. Thompson. *Compressible-Fluid Dynamics*. McGraw-Hill Inc., 1972.

[123] B. van. Leer. Towards the ultimate conservative difference scheme II. Monotonicity and conservation combined in a second order scheme. *J. Comput. Phys.*, 14:361–370, 1974.

[124] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *J. Comput. Phys.*, 187(1):22–46, 2003.

[125] V. Venkatakrishnan and D. Mavriplis. Implicit solvers for Unstructured meshes. *J. Comput. Phys.*, 105:83–91, 1993.

[126] G. Wang, F. Duchaine, D. Papadogiannis, I. Duran, S. Moreau, and L. Y. Gicquel. An overset grid method for large eddy simulation of turbomachinery stages. *J. Comput. Phys.*, 274:333–355, 2014.

[127] L. Wang and P. Persson. A high-order discontinuous Galerkin method with unstructured space–time meshes for two-dimensional compressible flows on domains with large deformations. *Comput. Fluids*, 118:53–68, 2015.

[128] R. Wang, P. Keast, and P. Muir. A comparison of adaptive software for 1D parabolic PDEs. *J. Comput. Appl. Math.*, 169(1):127–150, 2004.

[129] G. P. Warren, W. K. Anderson, and S. L. Thomas, James L .and Krist. Grid convergence for adaptive methods. *AIAA paper*, 1592:729–741, 1991.

[130] B. Webster, M. Shepard, Z. Rusak, and J. Flaherty. Automated adaptive time-discontinuous finite element method for unsteady compressible airfoil aerodynamics. *AIAA journal*, 32(4):748–757, 1994.

[131] D. Whitaker. Three-Dimensional Unstructured Grid Euler Computations Using a Fully-Implicit, Upwind Method. *AIAA Paper*, 93-3337:448–461, 1993.

[132] A. M. Winslow. Equipotential zoning of two-dimensional meshes. Technical report, California Univ., Livermore (USA). Lawrence Livermore Lab., 1963.