Titre: Title:	Moving Grids in Complex Configurations
Auteur: Author:	Mehdi Falsafioon
Date:	2016
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Falsafioon, M. (2016). Moving Grids in Complex Configurations [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/2173/

UNIVERSITÉ DE MONTRÉAL

MOVING GRIDS IN COMPLEX CONFIGURATIONS

MEHDI FALSAFIOON
DÉPARTEMENT DE GÉNIE MÉCANIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE MÉCANIQUE)
JUIN 2016

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

MOVING GRIDS IN COMPLEX CONFIGURATIONS

présentée par : FALSAFIOON Mehdi
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

M. GARON André, Ph. D., président
M. CAMARERO Ricardo, Ph. D., membre et directeur de recherche
M. GUIBAULT François, Ph. D., membre et codirecteur de recherche
M. TRÉPANIER Jean-Yves, Ph. D., membre
M. FRANÇOIS Vincent, Doctorat, membre externe

DEDICATION

To my parents : Sedigheh and Mohammad

To my elder brother : Ahmadreza

To my whole family

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my deepest gratitude to my supervisor Professor Ricardo Camarero. It has been an honour to be his Ph.D. student and to work under his supervision. I appreciate all his contributions of time, ideas, and advises to make my Ph.D. I could not have imagined having a better adviser for my Ph.D. study.

I also would like to express my special appreciation and thanks to my co-supervisor Professor François Guibault, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on research has been priceless.

I gratefully acknowledge Sina Arabi, my graduated colleague from Polytechnique Montreal, for his contributions and for providing the required information during my Ph.D. study.

My sincere thanks also goes to Robert Magnan, Paul Labbe, and Anne-Marie Giroux, who provided me an opportunity to join their team as intern, and who gave access to the research facilities at Hydro-Québec's research institute (IREQ). A special thanks again to Robert Magnan for being super supportive during my eight month internship in IREQ.

Last but not the least, I would like to thank my family : my parents and to my brothers and sisters for supporting me spiritually throughout writing this thesis and my life in general.

RÉSUMÉ

Un grand nombre d'applications industrielles reposent sur un écoulement qui est produit par le mouvement d'une frontière. Celles-ci présentent de grandes difficultés de modélisation tant du point de vue de la géométrie que de la physique. Malgré de nombreuses tentatives, la problématique de la représentation précise de l'évolution de la discrétisation du domaine demeure un défi. Dans ce travail, on étudie les aspects de la génération de maillages en utilisant une méthodologie de maille mobile dans le cadre des cellules en glissement.
Cette approche a été retenue après une analyse critique des méthodes présentement applicables pour aborder l'évolution temporelle de tels domaines de calcul, quant à la complexité des géométries, l'amplitude du mouvement relatif des frontières par rapport au coût et la robustesse de la méthode. Ces critères sont vérifiés globalement par plusieurs classes de méthodes avec les caractéristiques suivantes :

Adaptation des frontières : pour une représentation précises de la géométrie.

Un seul maillage pour tous les temps : la procédure de génération de maillage est appliquée une seule fois, donnant un maillage de référence à topologie fixe et un nombre constant de sommets. En évitant les différents procédés d'adaptation, les structures de données ne changent pas à chaque étape de temps et ceci donne lieu à une procédure robuste.

Mouvement du maillage : l'adaptation aux frontiéres mobiles est réalisée au travers des déplacements des sommets en utilisant diverses techniques d'optimisation ou de lissage qui en assurent la qualité et la validité. Grâce aux connectivités fixes, ces techniques peuvent être très efficaces, puisque seuls les sommets sont modifiés.

La méthodologie proposée hérite de la simplicité conceptuelle décrite ci-dessus, et apporte les contributions spécifiques suivantes,

Glissement de mailles dans l'espace physique : le modèle de maille mobile dans le cadre de maillage en glissement est représenté par un corps se déplacant au travers , et écartant les mailles d'un maillage de référence. En considérant, chaque maille comme une particule autour des frontières du domaine en mouvement, de manière semblable à un écoulement potentiel, et en appliquant une condition de glissement pour les déplacements des mailles, de grandes amplitudes sont possibles et les grandes déformations des mailles sont évitées. Il a été montré qu'en réalisant ces procédures directement dans l'espace physique donne lieu à des maillages de meilleure qualité et plus lisse.

La trajectoire : le glissement de la frontière à l'intérieur du maillage de référence repose sur

une technique de séparation et de fusion des mailles qui, dans la méthode de glissement de mailles originale, peut donner des topologies non valides. Afin d'assurer la validité du maillage tout le long du mouvement, ce maillage de référence est initialement généré autour d'une trajectoire pré-définie, représentée par une courbe dans l'espace physique. En appliquant l'idée de base de laisser les mailles glisser le long de la frontière du corps et en effectuant ces opérations dans l'espace physique plutôt que dans l'espace paramètrique, il a été montré que l'on obtenait globalement un meilleur recouvrement du domaine de calcul.

Mouvements de translation et rotation : les divers algorithmes et les structures de données associées ont été développés et appliqués à quelques configurations génériques simples de domaines et trajectoires. Le couplage avec des techniques efficaces de lissage pour le mouvement des mailles s'est avéré fonctionnel pour les mouvements élémentaires de translation et rotation, ainsi que pour le cas plus complexe de deux corps en mouvement relatif.

Maillages composites et déplacements complexes :une extension aumodèle d'un corps se déplacant à l'intérieur d'un maillage de référence a été réalisée où l'on déplace un maillage, solidaire au corps, à l'intérieur du maillage de référence. Il est alors possible d'effectuer des déplacements de maillages hybrides (par exemple, un maillage composé de quadrilatères près des parois et des triangles dans le champ lointain). Ceci permet également de déplacer des formes très compliquées et ce avec une bonne transition de densité et qualité de maille entre les parois et l'intérieur du domaine.

De plus, la méthodologie a été appliquée à des mouvements arbitrairement complexes, composés de translation et de rotation d'un corps se déplacant le long d'une trajectoire pré-définie

Couplage à un résoluteur Euler-Lagrange : afin de vérifier l'intégration globale des diveres techniques et algorithmes, ainsi que les structures de données mises en jeu, un couplage avec un résoluteur a été réalisé. Le schéma utilisé est une extension du schéma de Roe modifié pour inclure les lois de conservation géométriques. Des écoulements simples ont été simulés.

ABSTRACT

A wide range of industrial applications are based on an unsteady flow field which results from the motion of a boundary. These present considerable modelling difficulties with regards to both geometry and flows physics. Despite the numerous approaches addressed at such problems, the issue of the accurate representation of the evolving discretized domain in the context of numerical simulations remains a challenging problem. In this work, we will study the grid generation aspect using a moving mesh methodology based on a sliding cell framework.

This approach was selected after a critical review of currently available methods to handle such evolving domains on the basis of the complexity of the geometries, the amplitude of the relative motion of the boundaries versus the cost and the robustness of the method. These criteria are globally satisfied by a few classes of methods with the following characteristics:

Body-fitted : for an accurate representation of the boundaries;

One mesh for all times : the grid generation procedure is applied once, giving a reference mesh with a fixed topology and a constant number of nodes. By avoiding various dynamic mesh adaptation procedures, data structures do not change at each time step and this results in a robust procedure.

Grid motion : the adaptation to the moving boundaries is achieved via the displacement of the nodes using various optimization or smoothing techniques ensuring grid quality and validity. Because of the fixed connectivity, these procedures are can be made very efficient, since it only involves perturbing the vertices.

The proposed methodology inherits the conceptual simplicity of the above, and introduces the following specific contributions

Sliding mesh in physical space : the model for grid motion in the sliding mesh framework is that of a body moving through, and displacing the cells of a reference mesh. By considering each cell as a particle flowing past the domain's moving boundaries, analogous to a potential flow, and applying a slip condition for the cells' displacements, large amplitude displacements are possible and highly deformed cells are avoided. It was found that carrying out these procedures directly in physical space resulted in better quality and smoother grids.

Trajectory : the sliding of the boundary inside the reference mesh requires cell separation and reattachment procedures which, in the original sliding procedure, may result in the generation of invalid cells arrangements. To insure mesh validity throughout the

motion, this reference mesh is generated around a specified trajectory, represented as a curve in physical space. Applying the basic idea of allowing grid cells to slide along the body surface and carrying out these operations in physical space was found to yield globally better space-filling characteristics that working in computational space.

Translation and rotation : the procedures and associated data structures were developed and applied to several simple generic configurations of domains, body geometries and trajectory shapes. These were coupled to efficient grid smoothing algorithms for the cell motion and found successfull for both translation and rotation, and to the case of two objects in relative motion.

Composite grids and complex motions : instead of having a body slide through a reference mesh, an extension was developed whereby a rigid mesh, attached to a boundary, slides through the reference mesh. This allows a transition of cells with specific characteristics in the vicinity of a moving boundary to those in the outer reference mesh. This allows the motion of hybrid grids (for example, quadrilaterals close to the boundaries and triangles in the far field), and makes the motion of very complex shapes possible with good transition of cell density and quality.

In addition, complex motions, such as a body tumbling as it moves along a specified trajectory were realized as a combination of rotation and translation components, achieving general, almost arbitrary, motions.

Coupling with Arbitrary Lagrange-Eulerian solver : the final contribution, was to verify that the various procedures and associated data structure could be integrated into a flow solver. To this end, a Lagrangian Eulerian flow solver based on a generalized version of Roe's scheme was modified to include the geometrical conservation laws, and applied to a few selected moving boundary configurations.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1.1 Background

To achieve unsteady flow simulations with moving boundaries that incorporate the complete details of geometry as well as flow features, numerical algorithms with high fidelity are needed. This has led to the development of algorithms for simulating fluid physics, where dynamic grid generation for both viscous and inviscid regions play a significant role in the representation of the evolving geometry. While adequate approaches have been proposed for small motion amplitudes, the problem becomes considerably more complicated for large motion of bodies in close proximity. Despite numerous attempts addressed at this type of problems, available methods still impose significant restrictions on the complexity of the geometries, on the type of motion and changes in the topology.

Such unsteady phenomena occur in turbo-machinery applications which present considerable modelling difficulties with regards to both geometry and flows physics. Such flows are inherently unsteady due to the relative motion between the different components of the machine, e.g. rotor-stator or impeller-guide vanes interactions (Fig. 1.1). Furthermore, the flow is fully turbulent and spatially non-uniform. It is thus a formidable challenge to simulate accurately these complex flow fields occurring in such configurations.

As another example, Fig. 1.2 illustrates a valve configuration where the body motion is per-

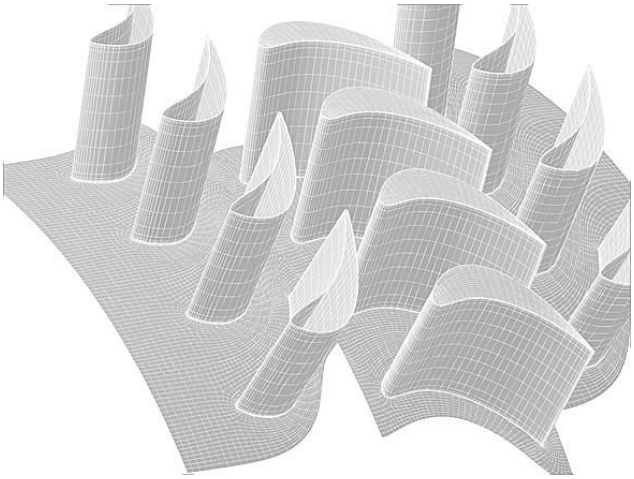


Figure 1.1 Relative motion in a turbomachine [1]

1. The photo is from `directindustry.fr`

iodic. Adapting the grid near the reciprocating valve in a time-accurate fashion is a critical task for this problem which requires knowing the distance between the valve and body.

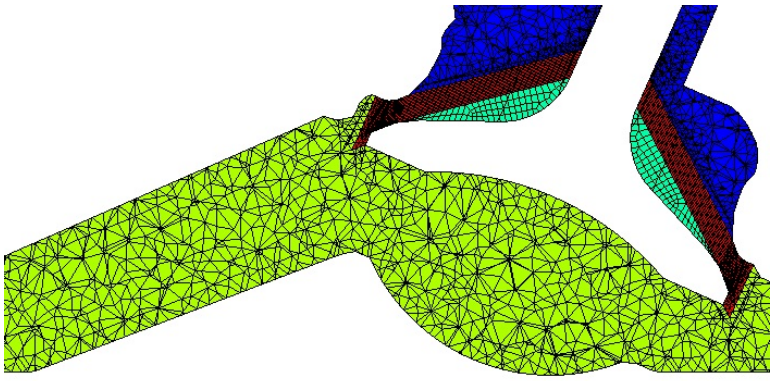


Figure 1.2 Geometry and periodic grid motion for an engine valve [2]

As it is known, at high Reynolds numbers, strong directional viscous stresses exist in a relatively small region of the flow domain adjacent to the wall, and this requires particular attention for the dynamic grid generation. This is further complicated when two boundaries come into contact or close proximity, resulting in topology changes and this is often cited among the most challenging problems in grid generation.

The most important characteristic of an unsteady flow with regards to moving grid generation is the amplitude of the relative motion of the boundaries. For instance, the simulation of an oscillatory flap (Fig. 1.3) can be handled successfully using the spring analogy technique for small displacements about a reference position. However, such methods eventually fail for very large motions because of the fact that the boundary cells remain attached to the boundaries, resulting in highly deformed or invalid grid cells.

Several moving mesh techniques have been proposed to handle large body displacements such as dynamic grid adaptation and overset grids. The basic idea is to make up for the expansion or contraction of the volume of the computational domain created by the body displacement. In the first case, this is achieved by the insertion, removal or displacement of cells in the grid domain, whereas in the second approach, different sets of grid cells (fixed to the boundaries) are moved over a background mesh while maintaining a sufficient overlap.

2. The photo is from `engines.polimi.it`

Both of these approaches require complex data structures to manage the mesh information, as well as time consuming interpolation of the solution data between different time steps.

To avoid these drawbacks, two novel advances have been recently proposed whereby the body in motion is displaced over a background mesh in such a way as to maintain a constant connectivity and a fixed number of cells. In the first instance, known as universal meshes (Rangarajan and Lew, 2012), the background mesh is deformed by displacing the grid cells intersected by the moving body and projecting the appropriate nodes onto the body's boundary. The second method (Arabi-Narehei, 2012) uses a background or generic mesh in computational space bounded by the non-moving boundaries and a slit representing the moving body. The body motion is managed in computational space, followed by a mapping of the computational mesh to physical space at each motion step, using an appropriate mapping operator. Managing cell movement in computational space presents some difficulties in handling physical domain features, especially in regions where boundary curvature varies rapidly. Also the length of the motion in this method was not controllable but restricted to the distance between two nodes on the direction of the motion.

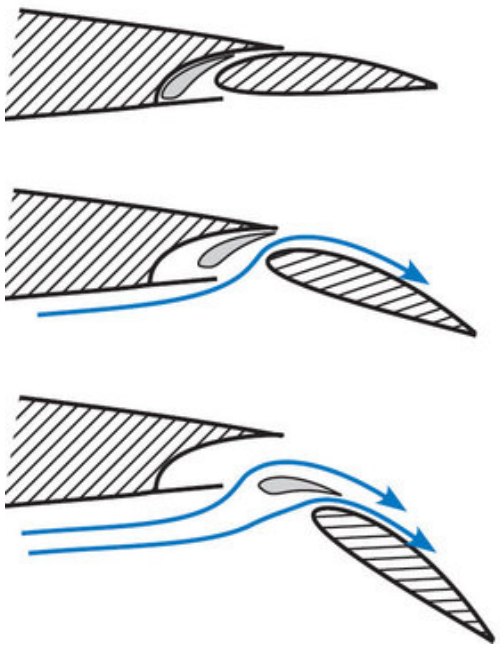


Figure 1.3 Large amplitude relative motion in a wing-flap configuration

These represent a new class of moving grid approaches, where a body moves through a reference mesh with a fixed connectivity and a constant number of nodes. In addition they are well suited for Arbitrary Lagrangian Eulerian (ALE) solvers and do not require interpolation of the solution at each time step.

1.2 Goal and Objectives

The goal of this work is to study a class of grid motion techniques which preserve the connectivity, the identity and the number of cells in a mesh. The approach is based on the sliding mesh methodology which allows arbitrarily large mesh displacement while maintaining good cell quality and space filling properties during the boundary motion.

This undertaking is structured around the following specific objectives,

1. extend the sliding methodology in order to make the motion displacement arbitrary and grid independent ;
2. develop grid management procedures in physical space in order to enhance the grid quality ;
3. devise efficient techniques to impart the body's velocity into the grid resulting in a smooth and valid grid ;
4. apply the method to a variety of complex configurations to assess its applicability to practical problems ;
5. couple the proposed global grid motion approach with an ALE flow solver in order to verify the grid management techniques and associated data structures ;
6. analyse the difficulties and consequently assess the functionality of the proposed approach.

1.3 Thesis Outline

Following the general context of grid motion induced by boundary displacements in fluid flow applications presented in this chapter, this dissertation is structured around the stated objectives and aims at the detailled presentation of a global methodology for the management of moving meshes for general configurations of body displacements in flow fields.

Chapter 2 presents a classification of the major approaches for moving grids and a critical review of the pertinent literature in the context of the stated objectives. This concludes with the rationale for the proposed methodology based on expected functional requirements. The proposed sliding grid methodology is reviewed in Chapter 3, together with detailled algorithms for the moving grid management in physical space for both, translating and rotating motion configurations. The chapter continues with a short description of the data structure. Various examples, illustrating the achieved objectives, are presented and discussed.

The issue of combining a given reference grid discretization (coarse) compatible with that (finer) required for an accurate description of the moving boundary is presented, and a solution based on composite grids is advanced in Chapter 4, along with relevant examples. The actual displacement of the grid nodes is achieved via a smoothing-type procedure described and assessed for both, efficiency and cell quality, in Chapter 5.

In Chapter 6, the performance of the proposed moving mesh methodology is applied to a few selected fluid flows using an Arbitrary Eulerian-Lagrangian (ALE) flow solver modified to include the geometrical conservation laws (GCL). This is not intended as a complete

validation but rather a verification of the proposed grid moving techniques and associated data structures coupled to a flow solver.

Finally, this document ends with conclusion, remarks and suggestions for future works.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction

Moving mesh generation techniques can be classified into two broad categories : body-fitted and non-conforming boundary methods.

Body-fitted methods : the geometry and its discretisation are explicitly defined, and, these can be further subdivided depending on the type of connectivity :

variable connectivity : these are essentially techniques whereby nodes/cells are dynamically inserted/removed to adapt the mesh to the evolving boundaries ;

fixed connectivity : the mesh is deformed to adapt to the changing geometry while maintaining the cells' identity (i.e. the connectivity and the numbers of nodes).

Non-conforming boundary methods : the geometry is implicitly defined, and it is not explicitly discretized. The motion is not that of a rigid body but rather an interface specified through a boundary condition governed by a particular physical phenomena such as phase change, a moving solid or a deformable body in fluid-solid interactions.

This classification is summarized in Tab. 2.1.

Table 2.1 Classification of moving grid methods

Body-Fitted						Non-conforming
Variable connectivity	Fixed connectivity					
Mesh adaptation	Mesh deformation				Multiple Meshes	Mesh Fixed
Dynamic Adaptation	P.D.E Laplacian	P.D.E. elasticity	Sliding Meshes	Universal Grids	Overset Grids	Immersed Boundaries

2.2 Mesh Motion with Connectivity Change

Early attempts in dynamic grid generation are essentially remeshing techniques, where the entire grid is regenerated, based on the new position of the boundaries (Zheng et al., 2005; Kwak and Pozrikidis, 1998; Saksono et al., 2007; Schneiders et al., 1992; Cristini et al., 2001). This approach was initially developed for adaptive flow simulations and then applied to the simulation of flows with moving bodies. This can produce meshes with high quality by defining an appropriate size function, such as equidistributing the error across the domain (Habashi et al., 2000). This approach is very expensive computationally, as it requires the

frequent application of the mesh re-generation procedure, as well as repeated interpolation of the solution between time steps for the flow solver. In addition, the interpolation process introduces a global error between time steps in the case of the Eulerian approach.

In order to avoid the time consuming process of remeshing the entire domain, one can target regions where the local element quality has deteriorated by forming a cavity which is subsequently remeshed. This approach has the advantage of reducing the amount of remeshing required, thus minimizing the regions where interpolation errors are introduced. In this way, the grid generation effort is considerably smaller than for global remeshing strategies.

This technique has been applied successfully in several dynamic simulations (Hassan et al., 2007) including bodies in relative motion where the mesh movement algorithm is modified to handle the highly stretched elements present in the boundary layers. The flow solution on the mesh at the previous time level is interpolated to the new mesh and the solution is advanced on the new mesh. However, especially in $3D$ cases, the generated void can be non-convex, for which remeshing with the desired quality can be a complicated task. In some cases, to satisfy the defined mesh sizing inside the voids, the surface of these voids has to be remeshed before applying the volume grid generation. This procedure increases the complexity of the method considerably.

This local remeshing approach can be applied on a smaller scale. Grid management operations, such as refinement, coarsening, edge swapping and edge collapse, can be applied locally on a grid. These can also be used for adaptation purposes, along with point insertion procedures and/or smoothing algorithms (Baker, 2005; Trépanier et al., 1993). If inverted elements are created during the mesh deformation, coarsening or refinement steps, then a single movement can be broken up into two smaller steps. If the situation persists, the step can be further refined.

These local mesh adaptation procedures can also be combined with various mesh movement techniques to form adaptation algorithms that will modify a mesh whose geometry and connectivity evolve in time. Specifically, in the case of moving grids, these mesh adaptive procedures can be used to fill/decrease the volume (area) increase/decrease due the displacement of the boundary. However, such remeshing approaches are computationally expensive. For some applications, essential physical features of the flow such as weak secondary shocks will be lost because of the inherent numerical diffusion resulting from the interpolation schemes in transferring data from one mesh to another (Hassan et al., 2007).

2.3 Mesh Motion with Fixed Connectivity

In order to overcome the major difficulties of mesh motion based on grid adaptation, another general approach is to couple the movement of all internal nodes to a prescribed boundary motion without modifying the mesh connectivity. Such mesh motion techniques where the connectivity remains fixed are called mesh deformation. Coupled to ALE flow solvers such meshing techniques combine simplicity with efficiency as the grid connectivity is generated once, and the nodal positions are updated following the boundary motion for each time step. In addition, the solution interpolation is avoided.

2.3.1 Mesh Deformation Based on Interpolation

Tranfinite interpolation is one of the earliest and simplest techniques for grid generation and can also be applied in a grid motion procedure. In this method, functions are constructed over a domain based on a given nodal distribution (Gordon and Hall, 1973). When used for grid motion purposes, the connectivity is fixed, and applying transfinite interpolation techniques, the internal nodes locations are computed for each time step. However, this method is limited to small amplitude of motion, and will fail to create valid grids when the deformed geometry becomes nonconvex.

Another interpolation method is based on radial basis functions presented by (Rendall and Allen, 2008) which can be applied to mesh motion while preserving the cells connectivity. The displacements of boundary nodes are propagated onto the interior nodes by the solution of a system of equations, involving only the boundary nodes. The method can handle small to moderate mesh deformations caused by translations, rotations and deformations, for both $2D$ and $3D$ meshes in fluid-structure coupling and mesh motion (Allen and Rendall, 2007; Rendall and Allen, 2009). This method is based on multivariate interpolation using radial basis functions and has several significant advantages. Primarily, all volume mesh and flow-solver type dependence is removed, and all operations are performed on totally arbitrary point clouds of any form. Hence, all connectivity and user-input requirements are removed from the motion scheme, as only point clouds are required to determine the dependence. In spite of all these advantages, the method cannot be used without modifications for large deformations (Arabi et al., 2014).

2.3.2 Mesh Deformation Based on PDE

Domain mapping using partial differential equations has been used extensively as a mechanism to generate and to smooth grids. Such approaches usually involve the solution of an

elliptic problem for bounded domains or a hyperbolic problem for external or open domains. Elliptic PDE's have the property that the solutions are inherently smooth while the hyperbolic PDE's allow particular characteristics to a grid e.g. orthogonality, but will propagate boundary discontinuities into the internal nodes.

Laplace Equations

Systems of Laplace equations with various diffusivity coefficients, have been used to generate and to smooth meshes (Löhner and Yang, 1998; Bau et al., 1997; Jasak and Tukovic, 2006). The grid generation problem is based on the heat conduction analogy with Dirichlet boundary conditions. This produces isotherms as grid lines that are smooth and non-intersecting. Their concentration and distribution in a given region can be controlled by adding source terms.

One of the features of using Laplace's equation is that the Jacobian is guaranteed to be positive as a result of the maximum principle for harmonic functions. Unfortunately, this only applies for the analytic solution and when the differential equation is discretized, the truncation errors may lead to invalid grids. This can also occur with the addition of forcing terms. Solutions to this equation satisfies the min/max principle, which means that the dependent variables on the interior of the domain are bounded by the values on the boundary of the domain (Thompson et al., 1998), thus insuring valid grids. While the behaviour of Laplace's equation is similar to that of the spring analogy, Laplace's equation with forcing functions (Poisson's equation) provides a better control on the mesh deformation procedure (Sorenson, 1980). These forcing functions must be defined to produce the desired grid shape near the boundaries. However, these have to be adjusted a posteriori, which decreases the automation capabilities and robustness of the method. Hence, devising a good system of Poisson equations is selecting appropriate forcing terms, which is thus equivalent to constructing valid grids in mesh deformation techniques. In particular, in an effort to simultaneously control both, edge lengths (which can be achieved by Laplace equation) and the normal mesh spacing, the use of bi-harmonic equations is proposed by (Helenbrook, 2003). In this technique, not only mesh spacing will be continuous at interfacial or periodic boundaries, but it also allows the simulation of problems with greater boundary deformation than Laplace's equation while preserving orthogonality.

Winslow Equations

Inverting the independent with dependent variables in Laplace's equations transforms the physical domain into the computational plane (Winslow, 1966), and the resulting system of equations, known as Winslow's equations, improves greatly the validity of the grid generation procedure. (Arabi et al., 2014) has presented a comprehensive study of the use of this map-

ping method for moving grids with an extension for unstructured grids. A triangular grid is generated in computational space and the moving body is represented as a sliding slit. This configuration is mapped to physical space using Winslow equations with different discretization methods (i.e. finite volume and finite differences) for unstructured grids. Comparisons with Laplace's equations and functionals show that Winslow's equation is consistently the most satisfactory automatic PDE method for grid smoothing and grid deformation purposes and will be used in the current work.

Spring Analogy

Another widely used method based on the solution of PDEs considers the mesh as a network of linear springs and consists in solving the static equilibrium equations for this network to determine the location of the grid points. However, the grid smoothness and regularity are lost when the grid is subjected to large motions (Batina, 1991). This has been improved by adding torsional springs for controlling the motion of grid points (Farhat et al., 1998). A detailed analysis presented in (Blom, 2000) shows that the torsional spring is necessary to retain validity of a moving Navier-Stokes mesh. The difference between vertex and segment springs to calculate the equilibrium edge lengths is presented in (Blom, 2000), and the segment spring method based on the modified stiffness has been applied for a pitching airfoil where the original spring analogy methods had failed. (Sheta et al., 2006) has also reconsidered the formulation to use solid structural elements in an effort to prevent cell inversion. (Murayama et al., 2002) proposed a method to relate the spring stiffness with the angle between the faces in order to deal with the cell inversion problem for large movement and deformation of surfaces. (Chen and Hill, 1999) developed an exterior boundary element method (BEM) solver that has a unified feature for the deforming flow field grid generation. Assuming that the mesh is to be embedded in an infinite linear elastic medium where the surface grid is treated as a deformable hollow slit, a pseudo elastostatic problem with semi-infinite elastic domain can be formulated. Treating the grid as an elastic solid (Johnson and Tezduyar, 1996; Baker, 2001; Nielsen and Anderson, 2002), which can be viewed as an extension of the spring analogy, improves substantially the robustness of the moving mesh in the sense of avoiding mesh crossing and negative volumes, however at the expense of a much higher computational cost.

Applications of the spring analogy for grid deformation can be also found for unsteady flow solutions (Venkatakrishnan and Mavriplis, 1996) and for aerodynamic optimization (Nielsen and Anderson, 1999) as well as for unstructured viscous grids (Kholodar et al., 2005). Recently, (Zhou and Li, 2013) have also proposed a new method based on a disk relaxation algorithm which is more efficient and robust than the semi-torsional spring method

and results in high-quality meshes especially near the deformed boundary.

Another attempt for solving large mesh deformation has been proposed by (Yang and Mavriplis, 2005) using linear elastic smoothing. One advantage of this approach is that it uses a variable elastic stiffness, inversely proportional to the cell volume, in order to preserve the mesh quality in viscous layers. In (Yang and Mavriplis, 2007), an optimization procedure based on the adjoint method for linear elasticity mesh deformation technique is presented. This technique seeks to compute an optimal distribution of the modulus of elasticity to enhance the robustness and extend the range of applicability of this mesh deformation technique for large displacement cases. While very robust for several engineering applications, this method has the same limitations as the Laplace equation and gives invalid cells for large motions, especially around high curvature regions or sharp corner points of boundaries.

In spite of the numerous improvements that have been proposed, spring analogy-based schemes eventually will fail for many engineering applications. When applied to the modelling of relative motion, these methods encounter problems in robustness for large deformations and refined grids resulting in grid crossing and negative volumes. In addition, they require the iterative solution of the static equilibrium equations to determine the new locations of the grid points at each time step. For three-dimensional viscous problems, this is still time consuming to propagate the motion of the boundaries to the volume grids in the flow field, although a major improvement over grid regeneration.

In general, the ability of the class of mesh deforming algorithms to handle geometric complexity is limited for a given fixed mesh size when compared with AMR or remeshing schemes. In practice, extensions of these mesh deforming algorithms in combination with local remeshing schemes have been proposed by (Hassan et al., 2000; Löhner et al., 1999; Johnson and Tezduyar, 1999). The amount of remeshing can usually be reduced to a minimum, which results in very efficient approaches but this conflicts with their robustness.

Another class of methods is the overset grid approach which uses a combination of grids fitted around the moving parts, set over a background mesh. Each mesh has a fixed connectivity resulting in efficient flow solver. The amplitude of the motion is taken by the relative motion between the body-fitted and background grids which must maintain a certain overlap.

2.3.3 Overset Grids

In static, as well as, in evolving domain configurations, capturing complex geometric features remains a challenging problem. This can be facilitated by using composite grids

which are particularly useful for structured, deforming schemes. Traditionally, such domain partitionning schemes are based on adjacent multiple blocks and are relatively successful for static grids. Applied to moving grids, these suffer from the same drawbacks as the single mesh deforming methods described in the previous sections, namely, highly distorted or invalid cells for large relative motion of the domain boundaries.

A particularly powerful approach is the use of composite grids where the domain decomposition is achieved via a system of overlapping, rather than adjacent or contiguous meshes. Initially proposed by (Steger et al., 1983) for static configurations, in this technique, named overset or chimera grid approach, a complex geometry is discretized by a set of independent, geometrically superimposed grids. These different grids are components of the overall mesh structure, where each mesh is constructed around a geometric part to reflect its specific geometric feature, and then it is imbedded into a global mesh as illustrated in Fig. 2.1.

The flow field is solved separately on each of these component grids with the inherent

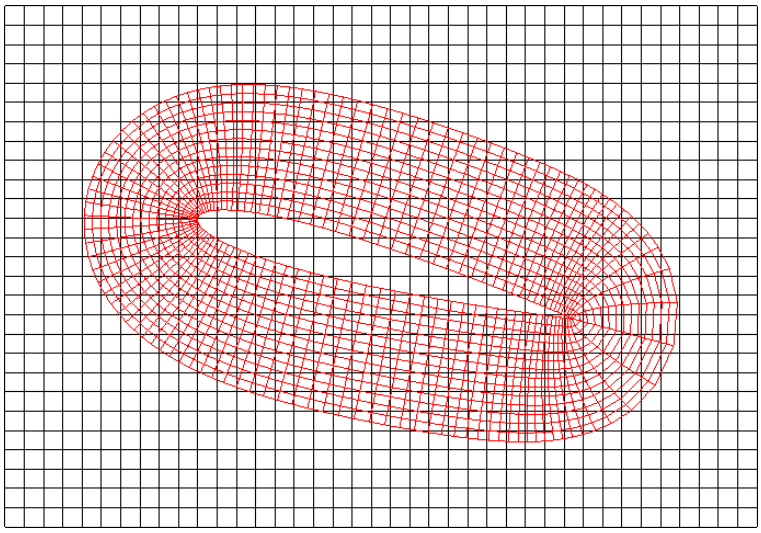


Figure 2.1 Grid overlapping for an airfoil with a Cartesian background grid [1]

advantage that the connectivity is fixed on a component-wise basis. This has been applied to numerous 2d and 3d flow problems ((Thompson et al., 1998),(Kao et al., 1994),(Meakin, 1997)). While the grid generation aspects have been considerably simplified, the flow solution requires a valid and efficient inter-grid exchange of the solution data. This is a rather complex task requiring expert know-how from the user, summarised in the following steps :

Grid generation : The domain decomposition is achieved by generating individual grid components : Usually a background or base grid is generated around the external

1. The photo is from `http://celeritassimtech.com/?page_id=15={Accessed:2015-09-16}`

boundary (a non-moving part), and several overset grids around imbedded solid parts (moving bodies) to resolve their geometric feature. These grids are superimposed in an arbitrary manner with the only requirement of a sufficient overlap to allow matching the solution across boundary interfaces.

Inter-grid communication : When solving for the flow variables in a given mesh, the nodes (cells) in a region of an imbedded mesh or those inside a solid body do not need to be computed. This is known as the hole-cutting procedure and consists in blanking these cells and identifying the boundary cells that lie along the hole boundary.

Intergrid communication is restricted to these boundaries, i.e. data is interpolated from the boundaries of the imbedded mesh to the base grid.

Interpolation stencils : This process involves finding a donor cell for the boundary cell in each grid and interpolating the solution from the donor cells to the boundary cells.

The greatest advantage of this method is that each component grid is simple to generate and can be structured, or unstructured, resulting in an overall mesh as an overset combination of these. However, one of the major drawbacks of this method is the complexity of the various operations necessary to manage the discretization (blocking, hole cutting, etc) as well as handling the overset-type flow calculation. The latter is rather involved and requires a significant amount of information to couple the various overlapping solutions. While several advances have been proposed to automate (partially) these tasks ((Wang and Parthasarathy, 2000), these are computer intensive and their implementation requires a high level of technical expertise from the user. A recent review of this approach can be found in (Pigeon, 2015).

Even using efficient search algorithms (Alternating Digital Tree) the searches for determining the hole boundaries and identifying the donor cells for the interpolation stencils can be very expensive for large grids. In steady-state solutions, where the geometries are fixed, these are applied only once. However, for moving bodies these procedures need to be applied repetitively as the grid around the moving boundaries traverse the computational domain since this changes the relative orientations of the grids. This requires that holes be re-cut and interpolation stencils be updated. In other words, while the connectivity within each of the component meshes is fixed and known, the connection between overlapping meshes has to be determined at each time step.

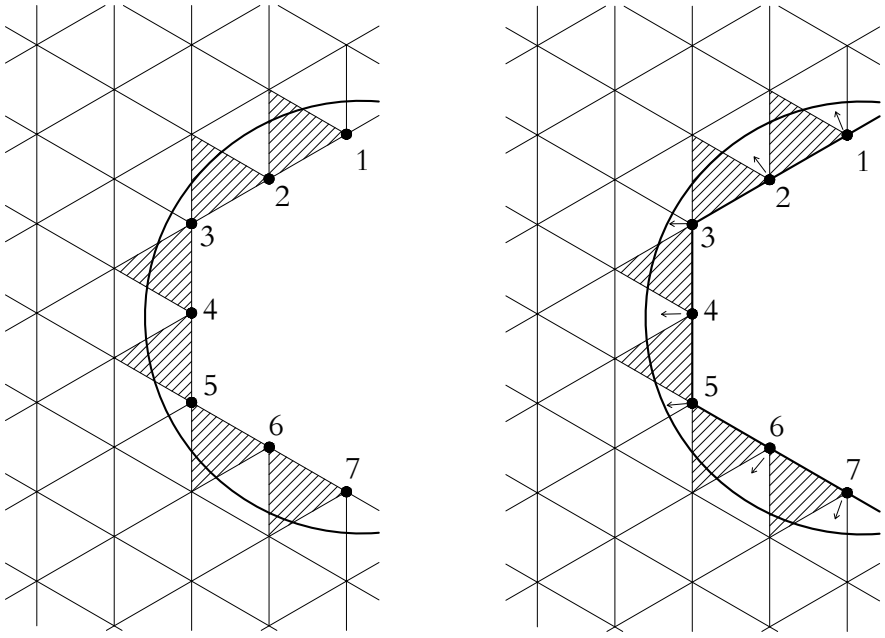
2.3.4 Universal meshes

A new class of methods to discretize an evolving domain occupied by a fluid, is to immerse the moving boundary in a mesh with fixed connectivity, and to body-fit the mesh to the body directly in physical space. This differs from remeshing or adaptive schemes in that

the mesh is generated once and subsequently deformed while keeping a fixed connectivity. It also differs from overset meshes in that it is a single mesh and does not require the intergrid interpolation. As the body moves through the domain, the mesh is refitted to the body at each time step by displacing some nodes and edges while keeping the same background mesh. Consequently, (Gawlik and Lew, 2015) named this a universal mesh for the fluid.

The method consists in constructing a mapping from triangles in a background mesh to ones that conform exactly to the immersed boundary. This is carried out in the following four-step procedure to adapt the background mesh by perturbing its edges and vertices, and adapt it to the moving boundary, as illustrated in Figs. 2.2 and 2.3.

Identify vertices in fluid : The intersection of the moving boundary with the background mesh allows to identify vertices belonging to the fluid region and outside.

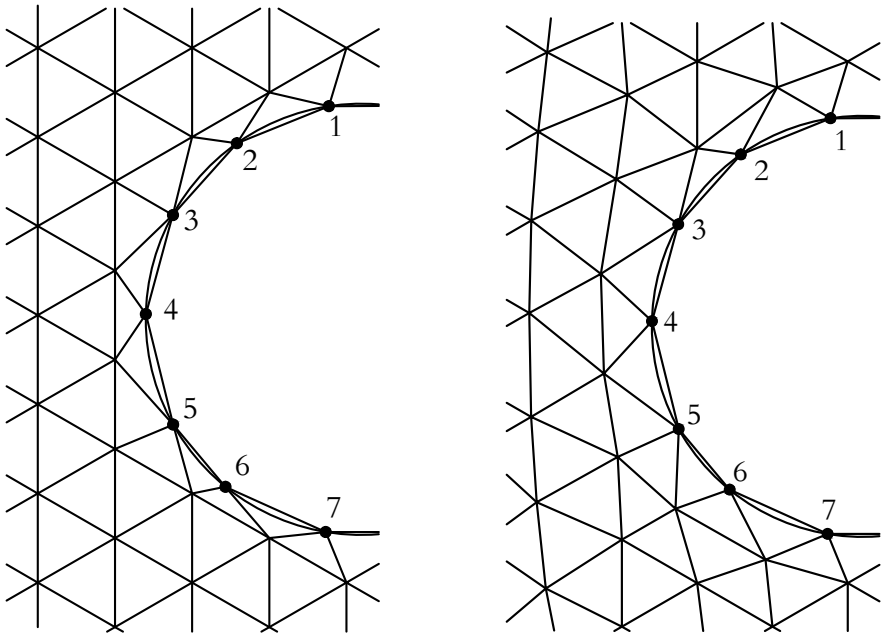


(a) Detecting positive vertices (b) Detecting positive edges

Figure 2.2 Mesh-boundary intersection

Identify positively cut elements : The intersection of the moving boundary with the background mesh generates positively cut elements, that is those with exactly one node in the fluid domain.

Projection : Edges belonging to this set of triangles having both vertices outside the domain are mapped (snapped) onto the boundary using the closest point projection.



(a) Projecting edges onto the boundary

(b) Smoothing the mesh

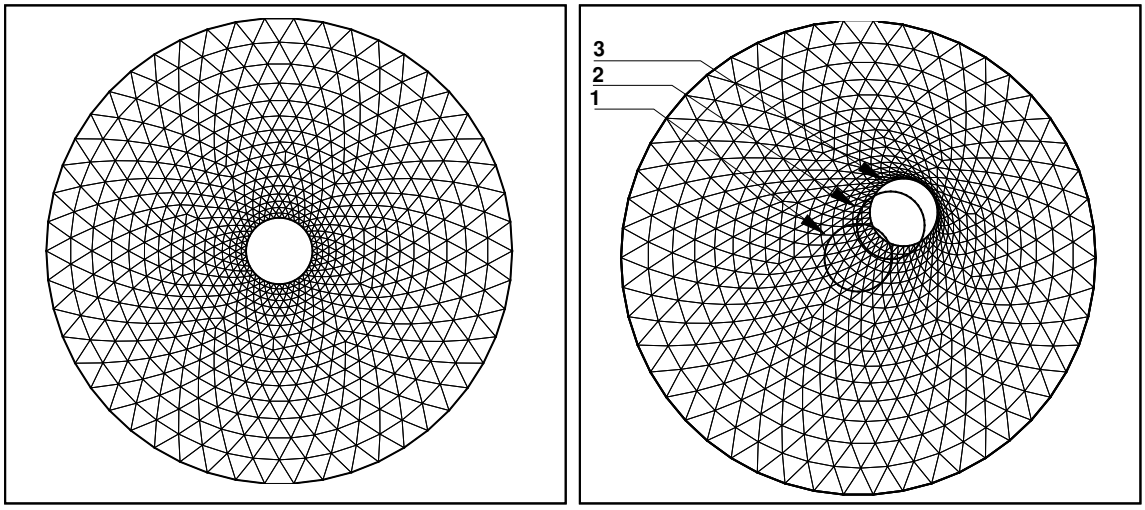
Figure 2.3 Mesh-boundary fitting

Smooth the mesh : Vertices within a given distance away from the boundary are relaxed to avoid very bad elements.

2.3.5 Sliding Method

In applying mesh deforming methodologies to simulate unsteady problems with large amplitude or relative motion of boundaries, the mesh is advanced in time, and at each time step, the coordinates of the internal grid points are updated according to the prescribed or computed movement of the boundaries. Such a deformation of the mesh takes place while the connectivity or topology remains fixed as shown in Fig. 2.4 for the motion of a circular boundary. After several time steps, the initial mesh becomes more and more distorted, decreasing the mesh quality and eventually leading to an invalid grid.

This is in general the case for the class of mesh deforming techniques discussed in the previous sections, which for large amplitude motion fail to produce a valid mesh because of the constraint whereby cells remain attached to the boundaries.



(a) Initial Mesh (b) Final mesh

Figure 2.4 Deforming the mesh by a body displacement (From (Arabi et al., 2014))

It is clear that coupling the mesh motion with adaptive grid techniques, skewed or flattened elements can be cured by applying edge swapping, node insertion/removal techniques as discussed in Section 2.2, and typically one obtains a new mesh as illustrated in Fig. 2.5 for the present example.

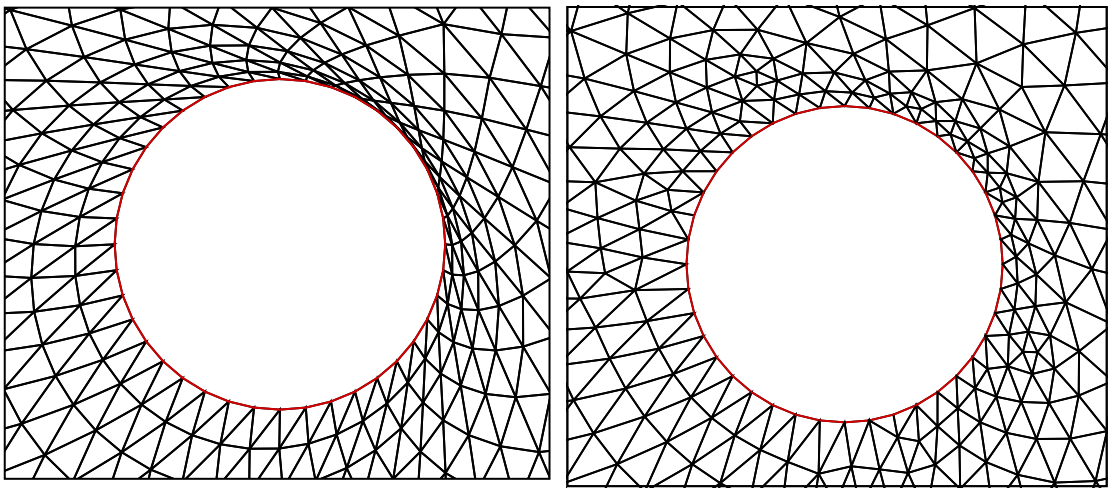


Figure 2.5 Swapping the flattened mesh to avoid mesh breakdown (From (Arabi et al., 2014))

However, devising such mesh topology modification schemes are complex and the continously changing grid topology requires interpolation of the flow solution at each time step,

in addition to the establishment of a criterion based on grid quality for the application of the curing step.

To overcome these problems, (Arabi et al., 2014) proposed an extension of the mesh deformation approach which allows cells to slide over the boundaries using a slip condition. In this model, each grid cell is considered as a particle flowing past the domain boundaries, analogous to a potential-like flow. The cells on the boundary are allowed to slip and follow a trajectory along the body as a streamline. As a result, cells will be deformed due both to the sliding motion and movement of the boundary while maintaining the same connectivity. With the extension of this slip condition, the global grid motion can be reformulated using the same basic PDE models such as Winslow's equations or the linear elasticity model described in the previous sections (2.3.1). Essentially, the basic approach consists of mapping an isotropic grid from computational space onto an arbitrary domain in physical space. This can be performed by the solution of a system of partial differential equations, where the target shape in the physical domain, Ω, is imposed by the body coordinates through the boundary conditions of the PDE solved in computational space, $\mathcal{C}$, as shown in Fig. 2.6.

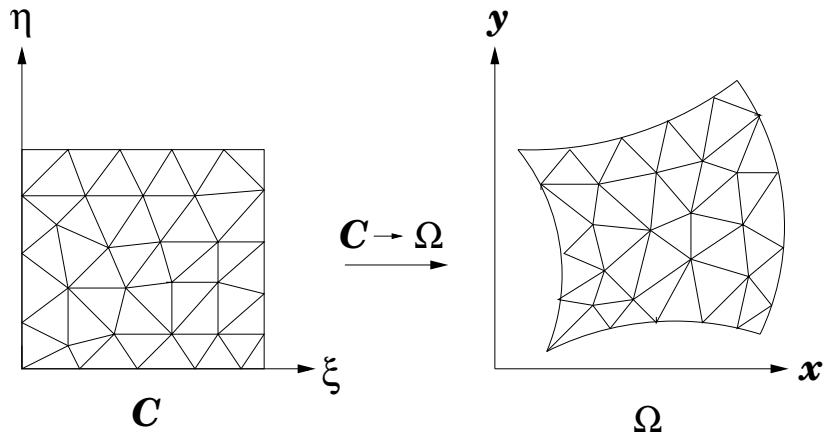
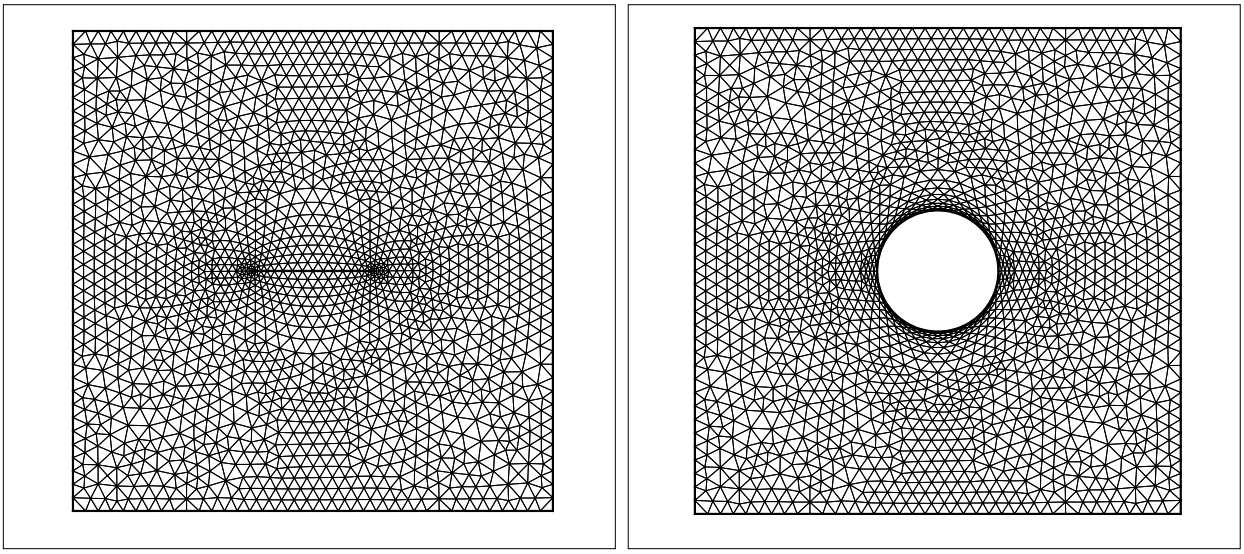


Figure 2.6 Mapping of a computational domain (ξ, η) to physical space (x, y).

This is illustrated by the example of a slit sliding through a mesh in computational space, mapped to an arbitrary shape in physical space (a circle in this case). It is noted that the physical mesh is created from computational mesh.

This method has been applied extensively and found to work adequately for slender bodies (Arabi-Narehei, 2012). In spite of numerous interesting features compared to other mesh motion strategies, this method suffers from a few short-comings. For instance, in the case of bluff body motion, this approach results in a stretching of the cells ahead of the body after just a few time steps. However, the greatest disadvantage of the method is mapping



(a) Grid in computational space (b) Mapping to physical space

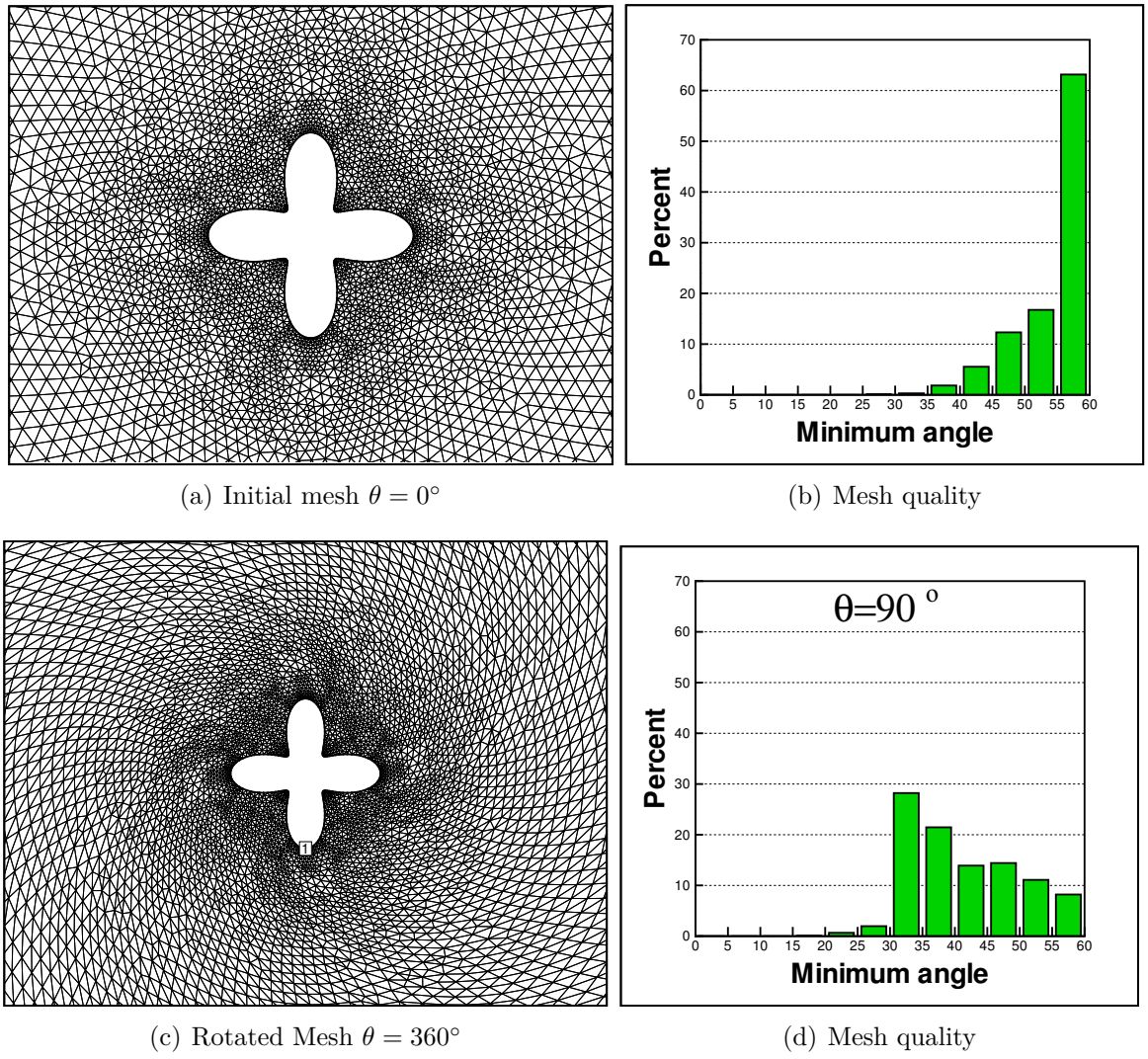
Figure 2.7 Generic mapping configuration for a body in translation motion : a straight line mapped to a circle(From (Arabi et al., 2014))

the solution from the computational space to the physical space which is obviously costly. Most importantly, the distribution of the nodes in physical space depends on the choice of the mapping in computational space. The consequence of this on the management of the grid motion is further discussed in Chapter 3.

Mesh coarsening and refinement can handle such situations but require choosing a proper time step according to the mesh size and boundaries velocity to preserve the grid validity.

2.3.6 Immersed Boundary Methods

All methods reviewed in the previous sections can be categorized as body-fitted approaches where the boundary is explicitly represented by the mesh, which follows time-evolving geometries in their movement. To avoid the cost and complexity of the grid generation step, another class of methods provides an alternative approach in which the domain boundary is immersed in a fixed background grid (Cartesian, curvilinear or unstructured) in a non-body conforming manner. The principal characteristic of these methods is the representation of the immersed solid via a modification of the numerical scheme in the vicinity of the immersed boundary, or by adding a forcing source term in the governing equations which brings the fluid velocity to zero, or by directly prescribing the no-slip condition, at the boundary. Known as immersed boundary, immersed interface, or embedded mesh methods,



(a) Initial mesh $\theta = 0°$ (b) Mesh quality

(c) Rotated Mesh $\theta = 360°$ (d) Mesh quality

Figure 2.8 Mesh deformation around a rotating four-petal rose

these differ in the specific details of how these additional constraints are implemented and how the fluid boundary conditions are enforced.

Pioneered by (Peskin, 1972) for fluid-structure interaction problems, these methods have been applied to numerous fluid flow problems with complex geometries and arbitrarily large boundary displacements ((Schillinger et al., 2012; Kempe and Fröhlich, 2012)), and transient compressible flows ((Wang et al., 2009)). As in general the boundary does not coincide with the mesh, the interface schemes required to satisfy the fluid boundary conditions can be either sharp or diffuse ((Frisani and Hassan, 2015). For sharp schemes, the forcing points are those grid nodes in the vicinity of the boundary, and the velocity on these are determined by

interpolation of the reconstructed velocity field in a manner that the corresponding boundary points satisfy the no-slip condition. This can lead to spurious oscillations. For diffuse schemes, the forcing points where the boundary force is evaluated are located on the boundary, not the mesh points, and the boundary force is then projected onto the grid nodes. The issue of sharpness is not entirely resolved and a compromise has to be reached concerning a diffuse imposition of the no-slip boundary condition.

Despite of the inherent ability of IB methods to handle arbitrarily complex domains with deformable boundaries, the greatest disadvantage of IB methods are their inability to selectively cluster grid nodes in the vicinity of solid boundaries, which creates difficulties in simulations of high Reynolds number turbulent flows (Sotiropoulos and Yang, 2014).

2.3.7 Meshless Methods

A family of methods called meshless methods has been introduced both for structured and fluid mechanics problems in two last decades. These new methods use the idea of a polynominal interpolant that fits a number of points minimizing the distance between the interpolated function and the value of the unknown points (Idelsohn and Onate, 2006). The first meshless techniques were proposed by Nayroles et al. (Nayroles et al., 1992). This was then extended to solve problems in structural mechanics by Belytschko et al. (Belytschko et al., 1994) and in fluid mechanics by Onate et al. (Onate et al., 1996a; Idelsohn et al., 2001; Onate et al., 1996b). Generally fluid mechanics problems with a moving free-surface are particular applications where the need of a particle method is clearer because this kind of problem needs a continuous update of the node connectivities, since two points, which are close to each other in a time step, may be very far from each other in the next one. Meshless methods do not typically need a conforming mesh, but just the connectivities between neighbor nodes in order to build the approximation functions. It must be noted that finding the node connectivities in meshless solutions may be as difficult as solving the mesh generation problem and, in most cases, the computing time to generate the nodal connectivities in meshless problems is of the same order than the most difficult mesh generation problem. As a result of this, the use of a meshless method is not relevant in most cases and the same problem can be solved with a mesh more efficiently (Idelsohn and Onate, 2006).

2.4 Critical Review and Proposed Approach

The aim of this work is to develop a moving mesh methodology that can handle boundary motion with large amplitudes easily while representing the geometry of the moving domain

exactly for prescribed boundary displacement. The methods commonly available to handle such evolving domains have been classified in Tab. 2.2 and discussed in the previous sections. The rationale for the choice amongst these is presented. In addition to the criteria for accurate domain representation during the motion, these various approaches will be assessed on the amplitude of the body motion, the cost/complexity and the robustness of the procedures.

In moving boundary configurations, the computational domain is continuously changing shape, and basically, this can be handled in one of two ways ; either the evolving domain is remeshed at each time step, or it is immersed in a background mesh and approximated within it.

In the first approach, the remeshing consists of the relocation and/or insertion/deletion of grid cells to fit the changing shape/volume of the computational domain. A variety of techniques have been reviewed and these require the construction of a new grid, fitting the evolving geometry at each time step. This yields a very good representation of the boundary, and also, usually a good quality grid. The cost can be very high but can be reduced by using local remeshing. However, in the latter case the mesh may not be smooth. Theses adaptive remeshing techniques are complex and moderately robust as the number of nodes, their location and grid connectivity change at each time step. In the second approach, the grid is trivially constructed, usually as a single cartesian structured mesh which remains fixed in time. The boundaries move separately from the grid and are not represented by the mesh, and this makes it imperative to approximate the immersed geometry sufficiently well over the background mesh. This is accounted for in a diffuse way by modifying the numerical scheme in the vicinity of the immersed boundary, or by adding a forcing term in the governing equations which prescribes the no-slip fluid velocity boundary condition. In this type of approach, the entire grid generation procedure is by-passed and replaced by substantial modifications to the solver algorithms. However elegant, these methods are rather complex and beyond the scope of the present work.

These approaches represent two extremes of the spectrum of the reviewed moving grids techniques : the simplest and most complex grid schemes. Consequently they score poorly as they are the furthest from the stated objectives in Section 1.2.

Thus, the remaining options to consider are the composite grid and mesh deformation methods which have the remarquable characteristic of having a fixed connectivity under mesh motion. In the first category, the multiblock approach provides good geometric representation but is difficult to automate. Furthermore, a given blocked domain will remain valid only for very small deformations. While over-set grids are capable of very good geometric representation and large amplitude motion, the cost and complexity of the intergrid communication

of the flow solution are simply too high in the present context.

The second category are the various classical mesh deformation methods which are very simple and robust but valid only for small amplitude movements. Large domain deformation can lead to cells with poor aspect ratios, and in more severe cases, element inversions can occur. The reason for this is that under deformation the nodes remain attached to the body as discussed and illustrated in Section 2.3.5. The universal mesh and the sliding mesh approaches avoid this in two different ways. In the former, the grid nodes of the background mesh in the vicinity of the moving boundary are relocated and those which are the closest are projected onto the geometry, so that the nodes on the body are not always the same and keep changing as the body moves. In a similar way, in the sliding mesh approach the nodes on the moving boundary are allowed to slide, so that they also keep changing. In both approaches this avoids the high cells distortion and/or inversion. Both preserve the connectivity of the mesh intact during grid motion allowing to impose fluid flow boundary conditions exactly without extra computational effort as required by the immersed boundary methods. In addition, both lead to higher efficiency and precision when compared to dynamic mesh adaptation or overset grid methods by avoiding the need for interpolation of variables as time evolves.

These evaluations, summarized in Tab. 2.2, are subjective on an absolute basis for individual criteria, but globally allowed to classify the various methods on a comparative basis. The final assessment is that the universal mesh and the sliding mesh approaches are comparable with an advantage for the latter due the simpler grid management procedure. So it was selected for an in depth study and to implement a number of extensions addressed at its shortcomings in Chapter 3.

Table 2.2 Evaluation of moving grid approaches

		Motion amplitude	Geometry representation	Cost and complexity	Robustness
Dynamic Remeshing	Global	good	very good	very high	good
	Local	good	very good	high	good
Mesh Deformation	Classical	small	very good	very low	very good
	Sliding	good	very good	low	good
	Universal meshes	large	very good	low	good
Composite Mesh	Multi-block	small	difficult	high	low
	Overset	large	very good	very high	small
Immersed Boundary	Body-fitted	large	small	high	difficult
	Approximated	large	difficult	high	difficult

CHAPTER 3 GRID MOTION

An approach is proposed to generate moving grids for large amplitude rigid body motion within the framework of the sliding grid methodology introduced by (Arabi et al., 2014). These new developments center around the grid management operations which are carried out in physical space rather computational space, and its application to practical flows for arbitrary and grid independent time stepping. These result in major improvements, addressing the shortcomings of the original methodology. Specifically, the introduction of a trajectory as a topological entity in physical space, and the underlying data structure designed to be compatible and coordinated with ALE flow solvers, insures valid moving mesh generation.

3.1 Introduction

Arbitrary motion of a solid body can be considered as a combination of translation and rotation which will be presented in Sections 3.2 and 3.6, respectively. The proposed approach for translation motion is called "zipping" which can be interpreted as a solid body moving through, and displacing the elements of an existing reference or background mesh whose connectivity remains fixed. To insure mesh validity throughout the motion, this initial mesh is generated around a specified trajectory, represented as a curve in physical space. The basic idea of allowing the grid cells to slide along the body surface is also used for rotational motion with specific procedures to manage the motion of grid cells for such configurations.

3.2 Sliding Method

Grid motion in the sliding method considers each cell as a particle flowing past the domain's moving boundaries, analogous to a potential flow. Fig. 3.1 shows the configuration of a body moving inside a grid and the adopted terminology to describe the method. The elements sharing a common edge ahead of the body at the leading node, are separated and slide along the upper and lower side of the body. As they reach the trailing node, they are reattached along their edges on the body which form a new common trailing edge, and are shed into the interior of the domain.

At each time step, the body position is updated, and the cells on the boundary are allowed to slip along the boundary as a streamline. As a result, cells near the boundary will be displaced and deformed due to both the sliding of neighbouring cells, and the movement of boundary. The mesh is smoothed using a PDE model after each time step.

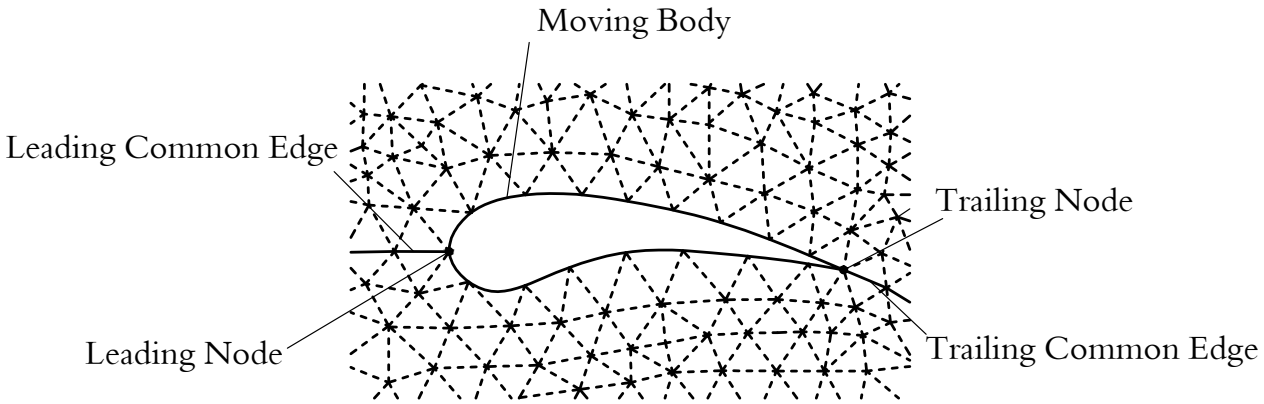


Figure 3.1 A moving body through an unstructured grid

The global procedure as originally proposed by (Arabi et al., 2014) consists in the following steps :

1. Generate the reference grid for a given body and domain configuration. The grid connectivity remains constant during the entire process ;
2. Identify the leading and trailing nodes [1] : as the body moves through the mesh, contiguous grid elements (cells) sharing the leading common edge will be separated and allowed to move along the upper and lower sides of the body, respectively. This will occur at the leading node, and a similar reattachment procedure will be applied at the trailing edge ;
3. Advance the body and slide the nodes/cells along it ;
4. Propagate the velocity of the body to the interior nodes by the use of a smoothing method ;
5. Repeat from Step 2 for the next time steps.

The method was applied extensively to test the procedure and found to work adequately for slender bodies. However, in the case of a bluff body, the approach resulted in a flattening of the cells ahead of the body after just a few time steps. This is due to two effects : the body motion piles and compresses the cells ahead of it, and the sliding on the body moves these out of the way. For bluff bodies, the former effect dominates and the sliding does not move the cells out of the way fast enough. The first and simplest cure presented in (Arabi-Narehei, 2012) was a restriction on the time step related to the cell size ahead of the leading node. This makes the moving procedure expensive and the control of time steps is not independent

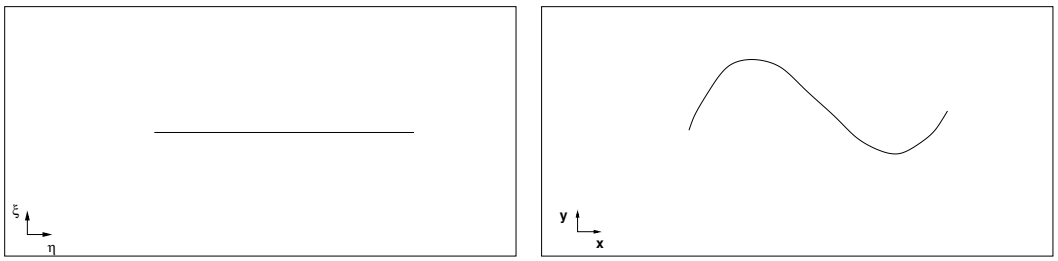
1. "Leading node" and "leading edge" terms are not strictly the same as the definitions in fluid mechanics

of the flow solver. The second solution proposed was to modify the model equation, so that the propagation of the boundary velocity into the domain is enhanced. This requires the addition of a proper forcing term to propagate the boundary's displacement to the interior nodes. The magnitude of this term can be based on a distance parameter which behaves like a distance field solution. These were found to be cumbersome and difficult to implement as they were too dependent on ad hoc procedures.

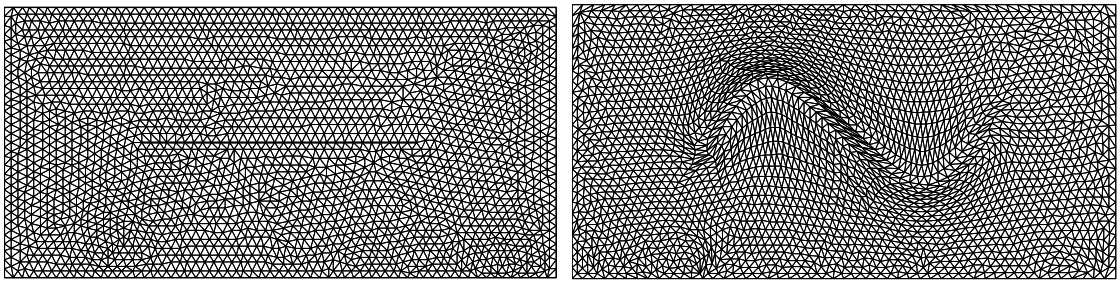
In spite of the unique capabilities of the sliding method compared to previous mesh deformation methods discussed in Section 2.4, several major drawbacks remain. These will be addressed around the following issues : 1. Reference Grid, 2. Cell Separation/Reattachment and 3. Mesh Motion and Time stepping

3.2.1 Reference Grid

The ability to generate an adequate grid motion depends on the mapping configuration. This is illustrated in Fig. 3.2(a) which shows the mapping of such a configuration from computational to physical space. In the original mesh sliding procedure, the grid is generated in computational space, around a slit representing the body and its path. This grid is then mapped to physical space as shown in Fig. 3.2(b)) from which it can be seen that the resulting grid in physical space does not account for the shape of the trajectory and, in general, will not yield a well distributed mesh.



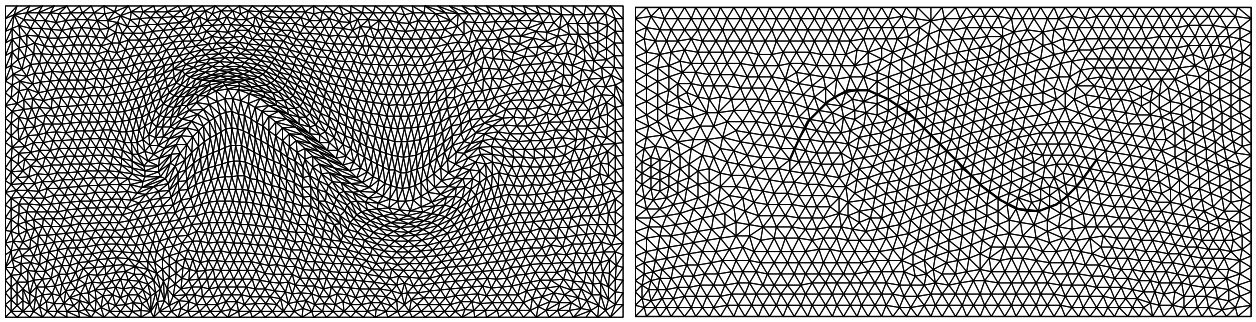
(a) Mapping configuration from computational to physical space



(b) Mesh mapped from computational to physical space

Figure 3.2 Mapped computational space meshes lack information about physical space

A comparison of this mapping approach with that of generating a grid directly in physical space is illustrated in Fig. 3.3. This shows that for the same configuration, the mesh generated directly in physical space, Fig. 3.3(b), gives a better domain representation than that obtained with a mapping procedure. As it can be seen, the mesh mapped from computational space to physical space is squeezed or stretched in some parts of the domain as the mapping operator is trying to fit the grid from computation space to physical space. A formal quantitative comparison between these two grids is presented in Fig. 3.4 which show that grid smoothness and quality are much better when the grid is directly generated in physical space. These grid quality criteria are discussed in detail in Chapter 5.

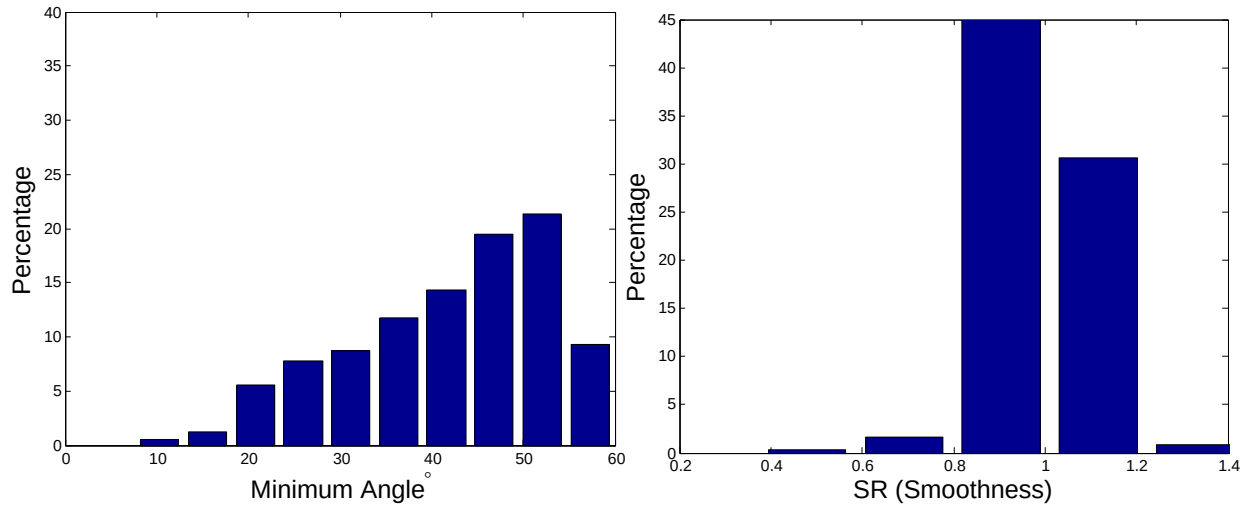


(a) Mesh mapped from computational to physical space

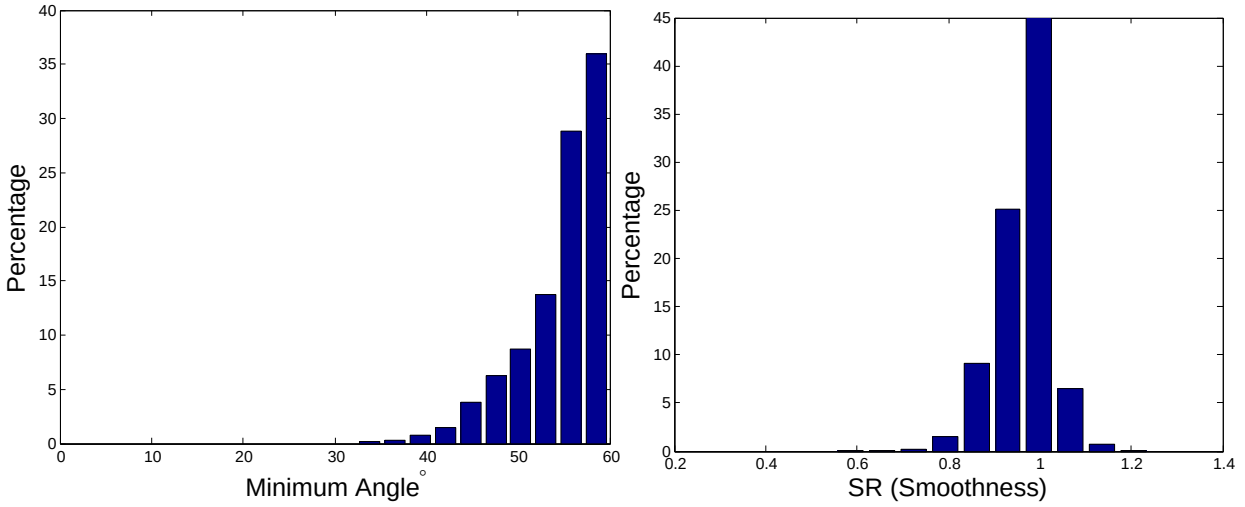
(b) Mesh directly generated in physical space

Figure 3.3 Grids directly generated in physical space have better space-filling characteristics

It can be concluded from these comparisons that meshes directly generated in physical space have better space-filling characteristics. This due to the fact that the mapping to computational space is generic and may not have all the information about the geometry of the configuration of physical space. In complex configurations with sharp curvatures, this leads to a mesh with regions which are too stretched or too squeezed. Generating the initial grid in physical space gives a better adaptation of grid to the geometry in each time step. Accordingly, it is the main reason that in the present work, the entire grid generation and moving procedure will be handled in physical space.



(a) Mesh from a mapping



(b) Mesh directly generated in physical space

Figure 3.4 Comparison of quality criteria for a grid mapped from computational space and a grid directly generated in physical space

3.2.2 Cell Separation/Reattachment

In applying the sliding grid procedure as illustrated in Fig. 3.1, the separation of the cells ahead of the body at step 2 may result in an invalid mesh depending on the valence or arrangement of the edges attached to the leading edge node. The process of separating two cells along their common edge is shown in Fig. 3.5 for different arrangements of the connectivity of the node of the candidate edge. Examination of the corresponding valence at the leading node before splitting shows that for a valence of three, an invalid mesh will result

after splitting. For example, for the third case in Fig. 3.5, the splitting process for two attached elements at the leading edge (elements $1-3-6$ and $1-5-6$), will result an overlapped grid which is obviously invalid. A configuration with a valence equal or greater than five will always result in a valid arrangement after splitting. In order to have an automatic valid

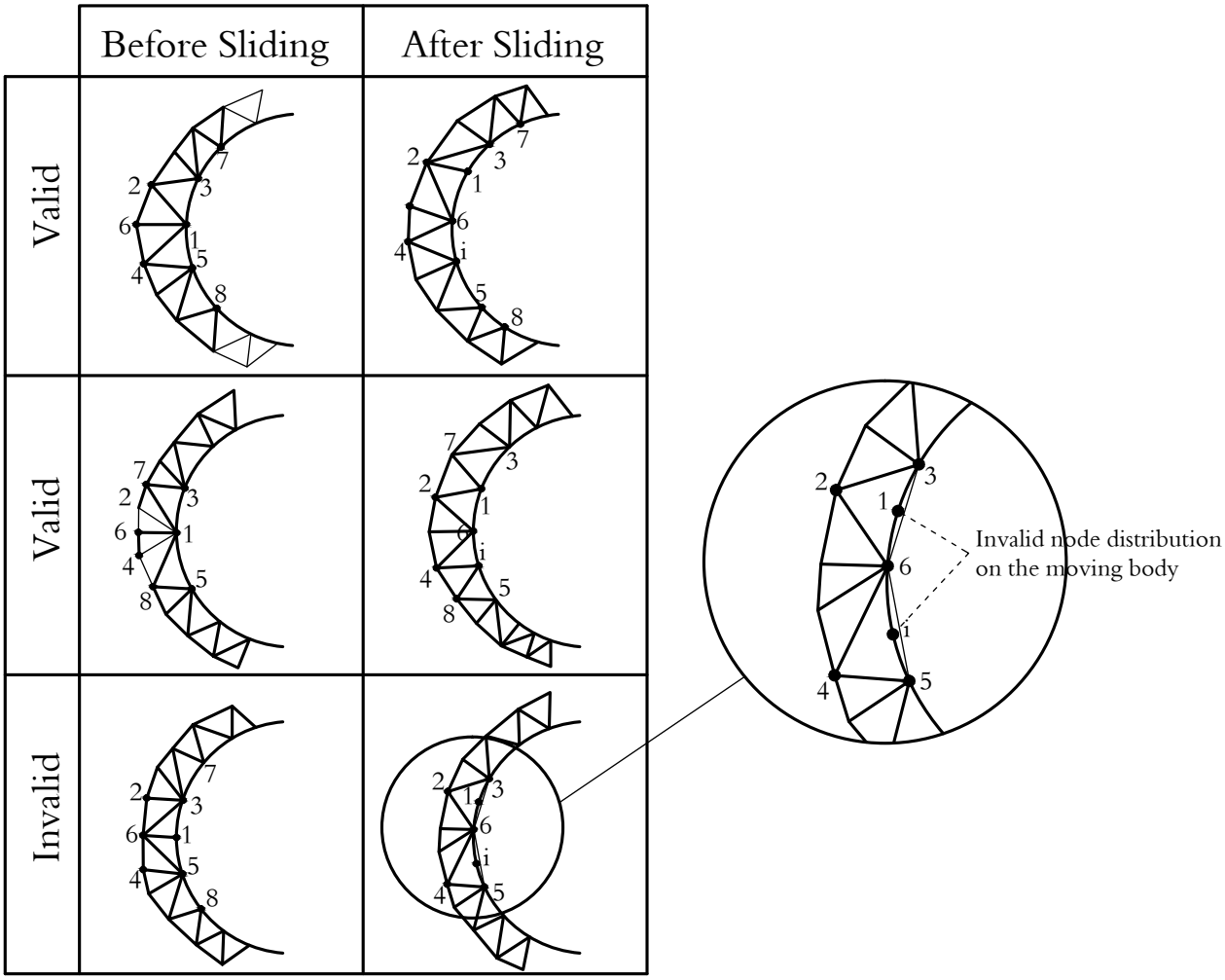


Figure 3.5 Valid and invalid arrangements of the edges ahead of leading node

arrangement of edges ahead of the leading node, (Arabi-Narehei, 2012) proposed to carry out the sliding procedure to the computational space. Thus, the reference grid is generated in computational space where the moving body is considered as a slit. At each time step, the slit is moved and the resulting configuration is mapped from computational space to physical space using a mapping procedure such as the Winslow equations. With this situation, one does not need to be concerned about the arrangement of the edge ahead of leading node. There is always a valid mesh after each time step, since the grid in computational space

remains fixed and unique. However, this has two drawbacks, the costly mapping procedure, as well as the fact that the reference grid in computational space may not have the best quality.

3.2.3 Mesh Motion and Time stepping

The body motion pushes and compresses the cells ahead of it and Laplace's equation is not very effective to handle the smoothing process after each time step. Using a mapping model based on Winslow's equations, and devising appropriate forcing terms based on a distance field was found to be very complicated, as well as computationaly intensive.

The body's position progresses from node to node. This snapping motion imposes a time step which is not controllable but defined by the distance between two neighboring nodes along the leading edge in computational space which is not easy to relate to the physical distance (and hence the time interval) in physical space.

3.3 Proposed methodology : The Zipping Method

The inherent features of the sliding method, invariant grid connectivity and sliding of the mesh cells along the moving boundaries, are implemented in a new methodology which is named the zipping method whereby the shortcomings discussed in the previous section, are removed by the use of the following extensions.

Reference grid The zipping method can be interpreted as the opening and stitching of a reference mesh along a pre-defined path, around a moving body, as shown in Fig. 3.6.

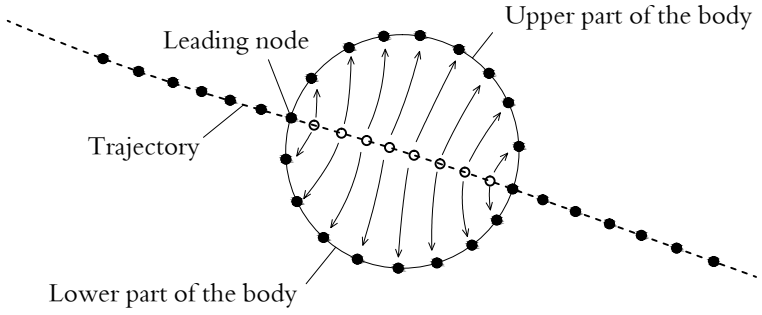


Figure 3.6 A moving body inflating a grid along a trajectory

Essentially, the body slides inside the trajectory and inflates/displaces the grid around it. Unlike the sliding method, the reference grid is generated directly in physical space and

is, topologically, multiply-connected, i.e. the moving body is a hole in the reference mesh. In addition, the entire grid management is carried out directly in physical space, avoiding the mapping procedure.

An explicit trajectory In the sliding method, to avoid rearranging the grid in the leading node in order to generate a valid splitting, it was necessary to use a mapping procedure from a mesh generated in computational space to physical space. In the zipping method, the reference grid is generated around the trajectory of the motion (Fig. 3.7), directly in physical space.

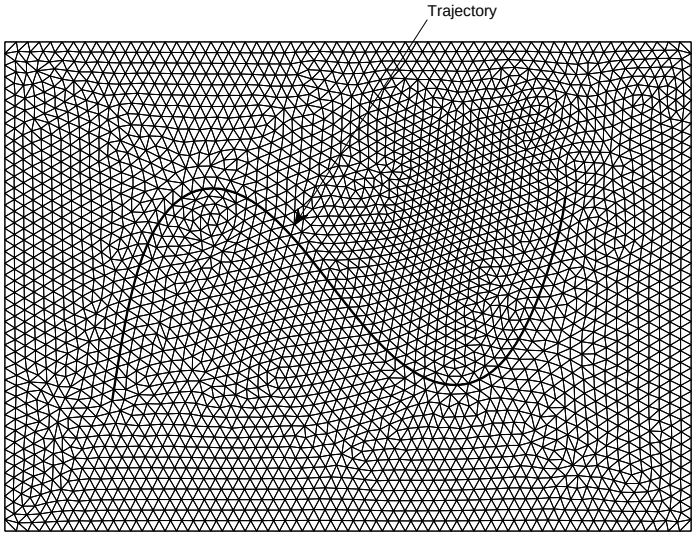


Figure 3.7 Reference mesh generated around a pre-defined path

As such, the valence or arrangement of the edges attached to the common leading edge for the cell separation/reattachment procedure is always valid. This is illustrated in Fig. 3.8 for six time steps for the case of a circle moving along a given trajectory. One can observe the motion and deformation of a general cell (numbered #1477) inside the domain, and more interestingly, the separation of cell #1600 from its neighbor and final reattachment and shedding into the domain.

As the topology of the grid is fixed during the entire motion, the edges are adapted to the specified motion and the arbitrary displacement of the body along the trajectory is satisfied with this strategy. This allows to overcome two major issues, i.e. controlling time steps as well as having a valid arrangement of the edges at separation and reattachment.

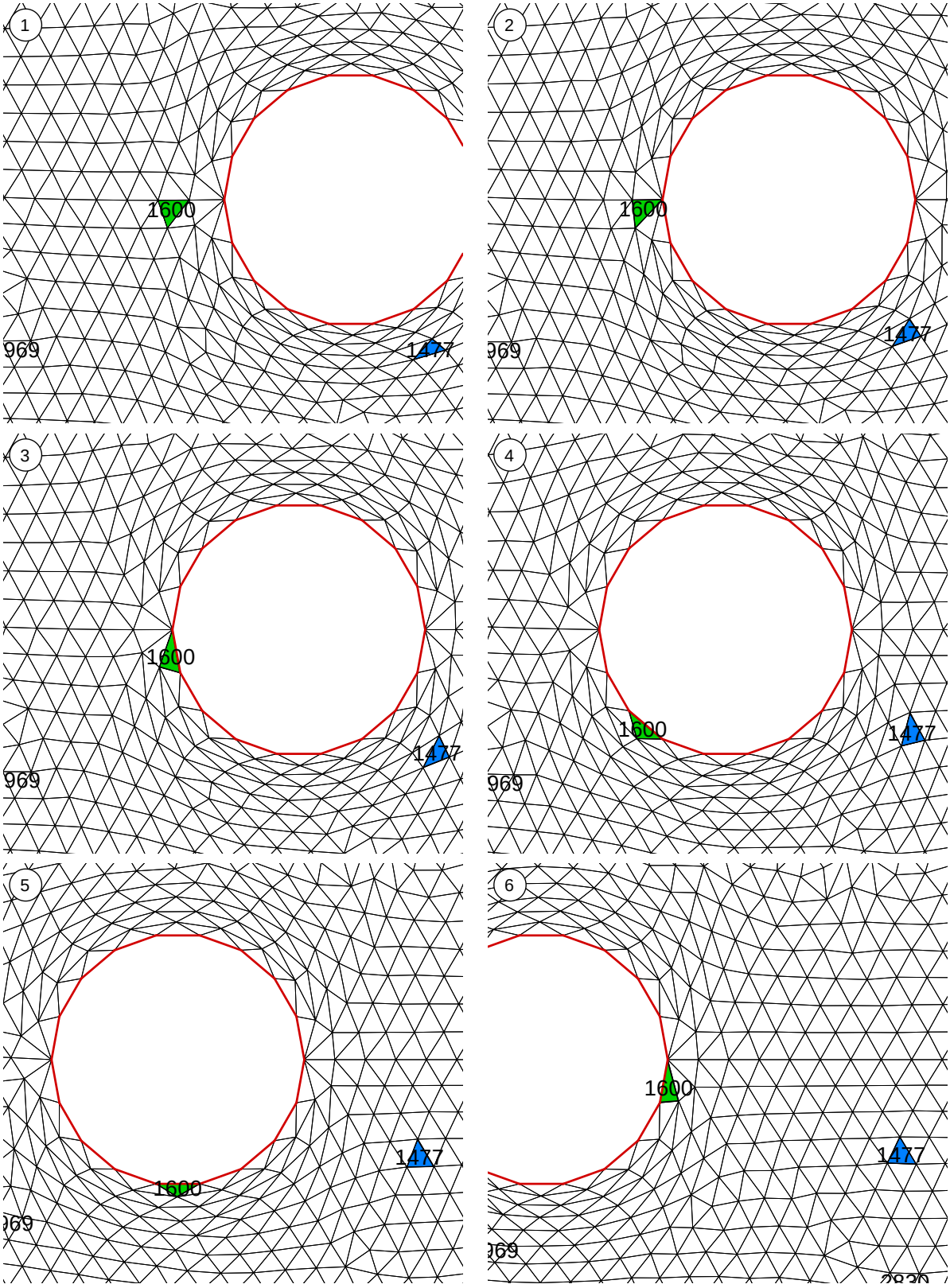
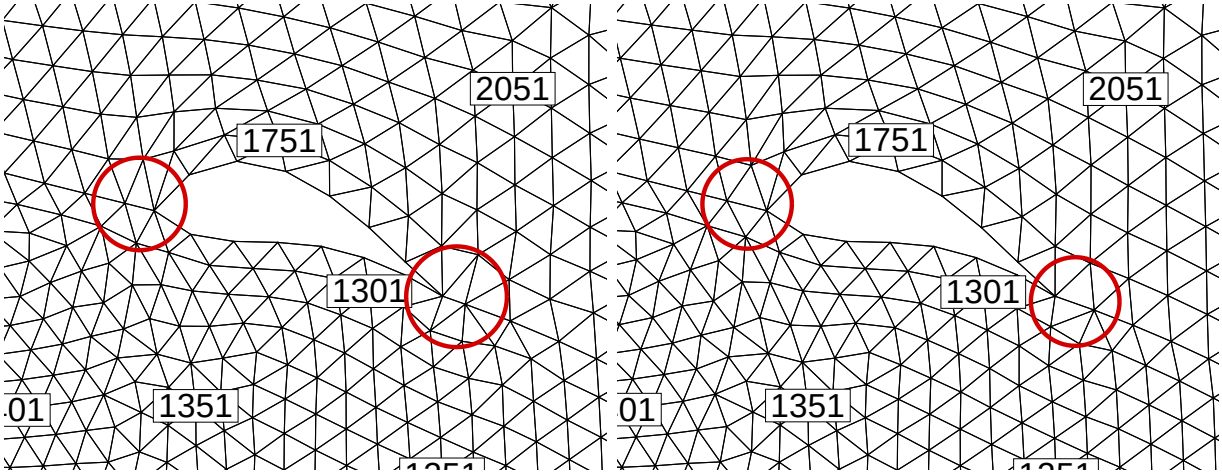


Figure 3.8 Illustration of proposed methodology for sliding grid motion

Time stepping As mentioned in Section 3.2, the displacement of the moving object in the sliding method is constrained to the distance between the leading node and its neighbour along the common leading edge. Defining the trajectory gives the possibility of having arbitrary length of the motion steps, as the trajectory can be rediscretized with the same nodes at each time step, based on the new position of the body. With the proposed method, this can be managed in three different ways by manipulating the nodes on the trajectory. The first and simplest way is to move the body along the trajectory by snapping the body's position to the neighbor nodes on the trajectory. This is identical to the sliding method except that the motion is carried in physical space. In this way, the number of nodes on the upper and the lower parts of the body is always fixed and the displacement of the motion is not controllable but restricted to the length of the leading common edges at each location.

In order to have arbitrary time steps independent of the grid, nodes on the trajectory can be redistributed after each displacement. This allows full control on the length of motion in each time step that makes the method applicable to a flow solver where the time steps and consequently displacements are calculated. In this way, the body is moved on the trajectory independently of the grid, and the body and the trajectory are rediscretized based on the new distribution of nodes ahead, after and on the body. This yields a smooth distribution of the nodes on the body during each time step (Fig. 3.6). The position of the two nodes, on the trajectory and outside the body, closest to the body, will be adjusted respectively to the leading node and trailing node of the body (Fig. 3.9(a)). In addition, the nodes on the trajectory ahead and after the body can be redistributed at each time step (Fig. 3.9(b)).



(a) Without re-positioning the nodes (b) Re-positioning the nodes

Figure 3.9 Effect of redistribution of the nodes on the trajectory ahead and behind the body

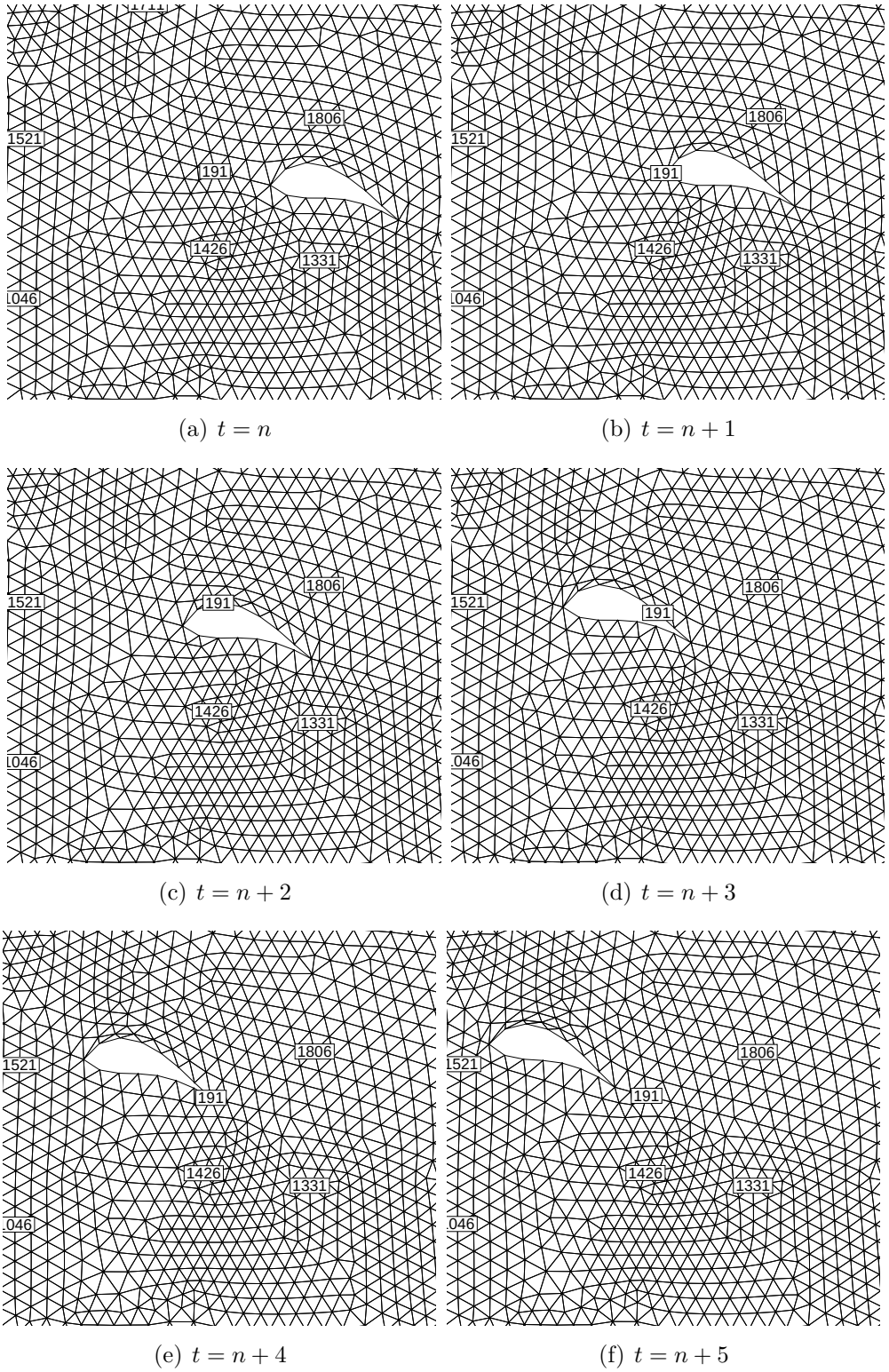
This helps to have a smoother node distribution on the trajectory and consequently gives a smoother mesh in each time step. This also avoids node squeezing at the vicinity of leading node and trailing node as is shown in Fig. 3.9(b). Although this causes some side effects on the flow solver, which are discussed in Chapter 5, it helps to have a smoother grid at each time step.

To illustrate the evolution of the grid as it slides past the body, Fig. 3.10 shows the tracking of a given set of nodes during three consecutive time steps. As it can be seen, node #191 at time $t = n$ is on the trajectory ahead of the moving body, then at the next time step ($t = n + 1$) it is as the leading node. At $t = n + 2$ and $t = n + 3$ the node is sliding on the body, and at $t = n + 4$, it reaches the position as node on the trailing edge. Finally at $t = n + 5$, the body has completely passed node #191.

Contrary to the sliding method, where the body motion has to adapt itself to the grid, in the present approach, the nodes on the trajectory are adapted to the new position of the body. Accordingly, the number of nodes on the body can vary at different time steps since the motion is arbitrary. This leads to have different number of nodes on the body in each time step which shows that the displacement is not restricted to the distance between two consecutive node on the path of the motion. Fig. 3.11 shows such a condition where the number of nodes on the body are different in two consecutive time steps ; six nodes for the time step t, and seven nodes for time step $t + 1$.

Grid Smoothing At each time step the grid is moved and smoothed by an appropriate smoothing operator. Since the modifications to the grid between two time steps are small, the smoothing process is fast and only a few iterations are required. Based on the topology and the curvature of the moving body, the employed smoother can be different. For a slim body, a simpler grid smoother, e.g. a barycentric method, can be employed, however for bluff bodies (with larger curvatures) a more effective grid smoother is needed e.g. Winslow equations. This is detailed in Chapter 5 where Winslow's equation is presented to have a well-smoothed grid in each time step.

Grid velocity Body motion through the grid imposes a velocity to the grid as the moving object pushes the nodes along its way during the zipping procedure and then a smoothing operator is applied in order to move the internal nodes to smooth the grid. This implicitly gives a velocity to the internal nodes and consequently is needed to be taken into account for an ALE flow solver. This is discussed in full detail in Chapter 6 where the proposed method is applied with an ALE flow solver.

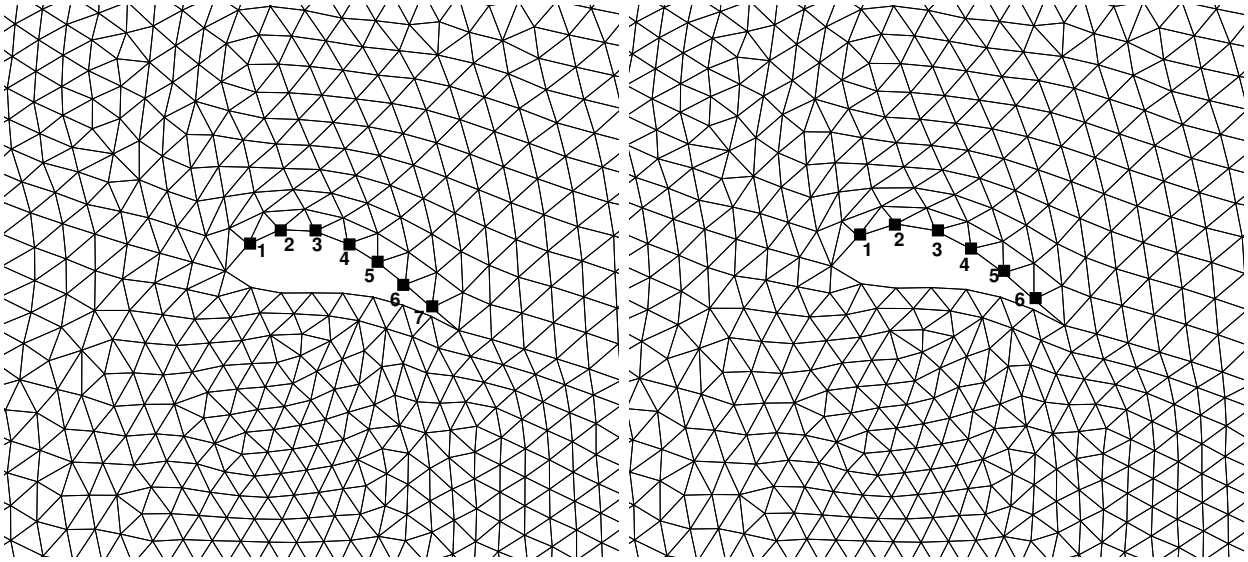


(a) $t = n$ (b) $t = n + 1$

(c) $t = n + 2$ (d) $t = n + 3$

(e) $t = n + 4$ (f) $t = n + 5$

Figure 3.10 Motion of Node #191 at six different time steps



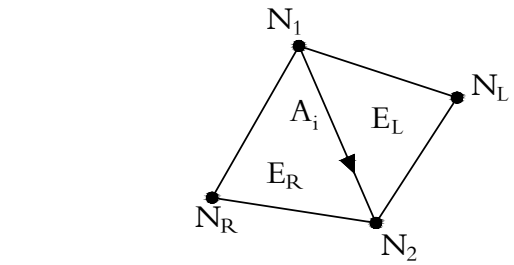
(a) $t = n$ (b) $t = n + 1$

Figure 3.11 The number of nodes on the body changes at two different time steps

3.4 Data Structure

In order to automate the process through the use of a computer code, one needs to provide a data structure that can identify and store the mesh components associated with every node of the discretization of the domain. In addition to this, a data structure is required to update certain properties of the grid such as the connectivity, edge and node properties and boundary conditions. This data structure needs to be designed comprehensively in order to cover all the needs related to the translation and rotation motion.

The data structure is edge-based and relates four components : nodes, edges, elements and boundaries structured around an edge as shown in Fig. 3.12. The edge is an oriented topological segment from node N_1 to N_2, shared by the left and right elements, E_L and E_R, and left and right nodes, N_L and N_R. The link to the geometry is through a reference of N_i to $x_i(N_i)$ and $y_i(N_i)$, the physical location of the nodal coordinates (x and y). The link to the computational domain is a reference to an internal or a boundary segment. In addition to this, the data structure maintains the connectivity information for each node, on node-to-node and node-to-edge basis as required by the various procedures such as the grid motion and the Euler flow solver which use fluxes passing through each edge in a finite volume scheme.

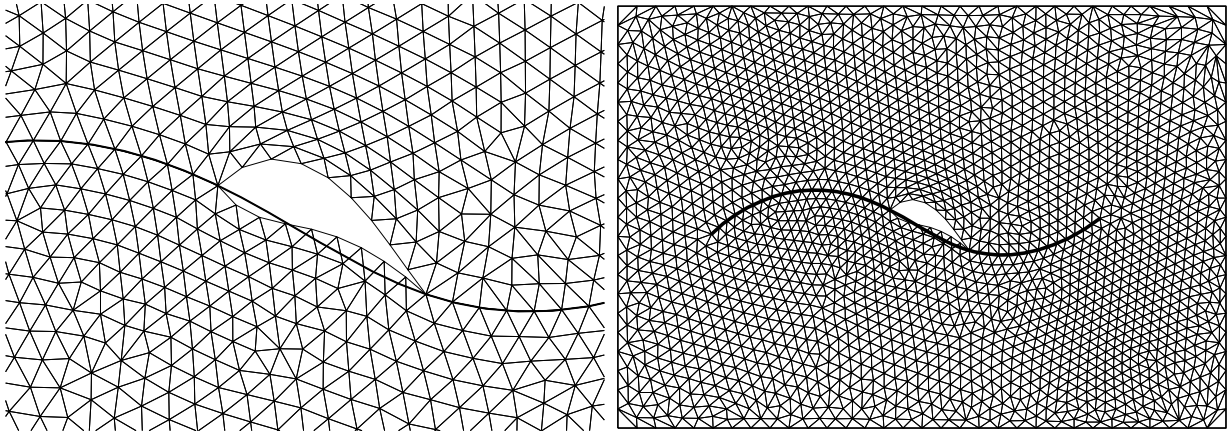


A_i	N_1	N_2	E_L	E_R	N_L	N_R	BC

Figure 3.12 Information stored for an edge

3.5 Overall Procedure

The trajectory on the zipping path has two sides which are "inflated" as the body slips through the grid. This requires that the body be represented by its upper and lower sides which connect at the leading and the trailing nodes, respectively. Therefore these two parts need to be specified, together with the trajectory at the beginning of the procedure as shown in Fig. 3.13.



(a) Close-up view (b) Global configuration

Figure 3.13 Body moving through a mesh along a given trajectory using the Zipping method.

The core of the algorithm of the zipping method consists in updating the data structure at each time step to reflect the location and neighborhood of each node and cell. Fig. 3.14 shows the terminology used to describe the basic configuration of a moving object set on a trajectory.

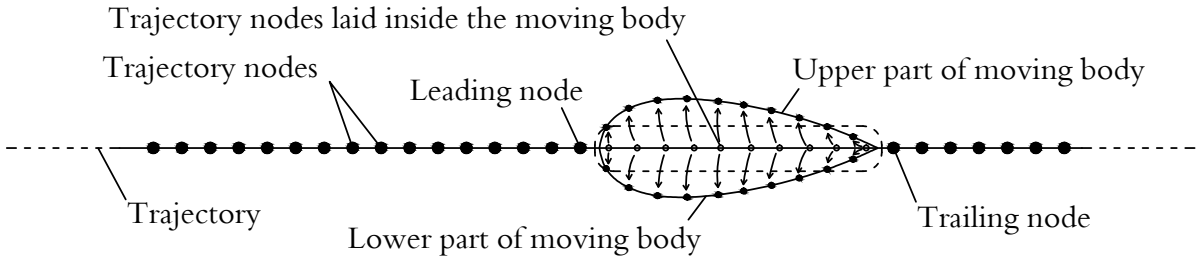


Figure 3.14 Terminology used to describe the translation procedure

In Fig. 3.15 the body is shown on the reference grid before it is "inflated". The edge and node information is shown for a part of this figure. Those nodes on either side of the trajectory are then transformed to the upper and lower part of the moving object. For example, node

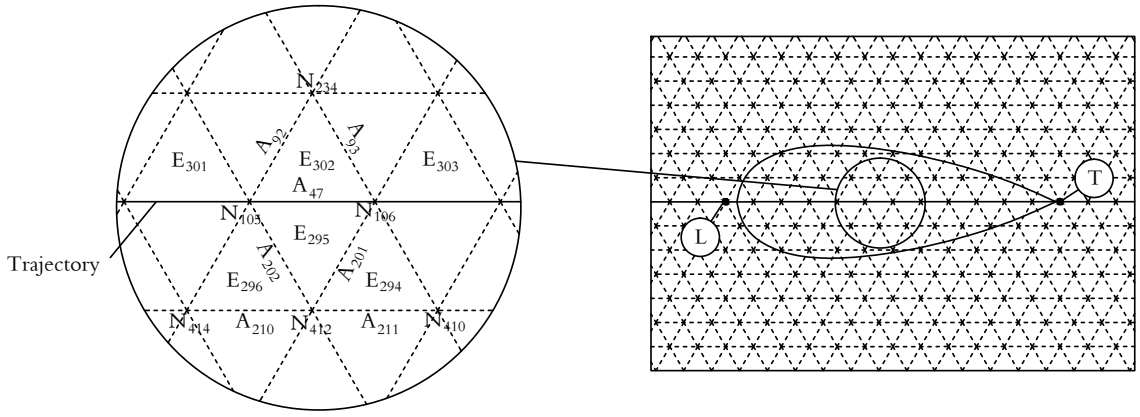


Figure 3.15 Moving object inside a grid before zipping process

N_{105} and node N_{106} give two new node N^+_{105} and N^+_{106} corresponding to A_{47} respectively. This procedure is carried out on the reference grid around the trajectory at each time step. Since the reference grid is created once at the beginning, there is no need to keep track from the previous time step. At the next time step, the body is displaced to its new position on the trajectory based on the calculated time step. The nodes on the trajectory will be inflated to the new position of the body and finally, the data structure will be updated. In Fig.3.16, those specific edges that are to be updated at each time step are shown with continuous lines and those edges that remain untouched are represented with dashed lines. The updating procedure is illustrated in Figs. 3.17(a) to 3.17(c). As each part of two elements with a common edge on the trajectory and laying inside the moving body are required to be separated for the inflation process, two set of edges will be generated to represent the upper and lower parts of the body. In the current work, for the upper part, the edges on

the trajectory and inside the moving body are transferred to the upper part of the moving body and a new set of edges are created in order to represent the lower part. For instance, in Fig. 3.15, edge A_{47} is transferred to the upper part (Fig. 3.17(b)) and is duplicated for the lower part giving the edge A_{n+4} as the mirror edge (Fig. 3.17(c)).

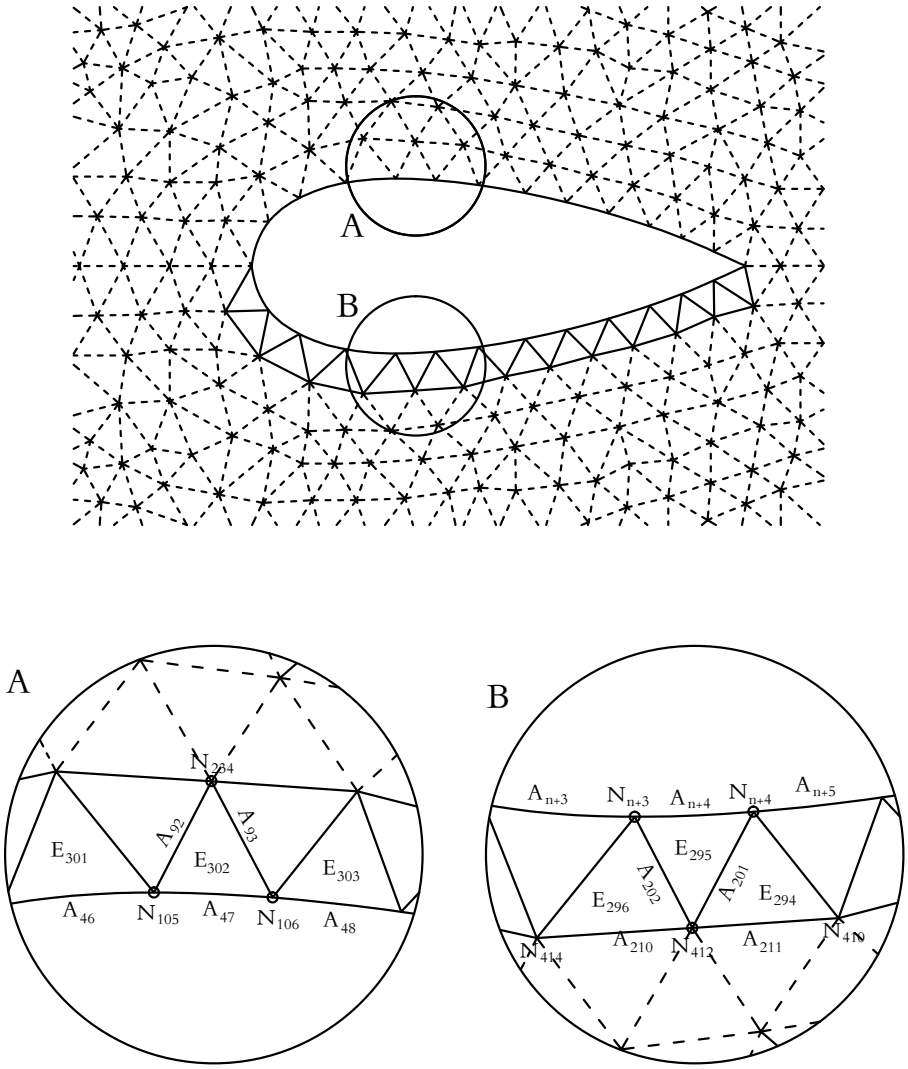
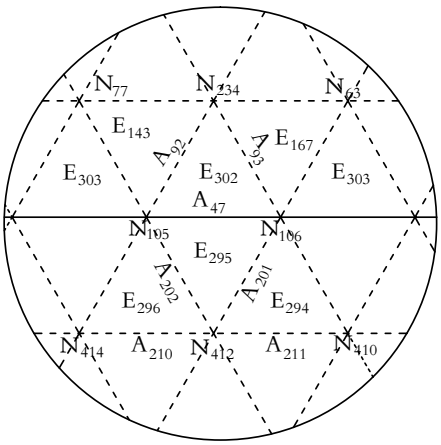


Figure 3.16 Moving object inside a grid after zipping process

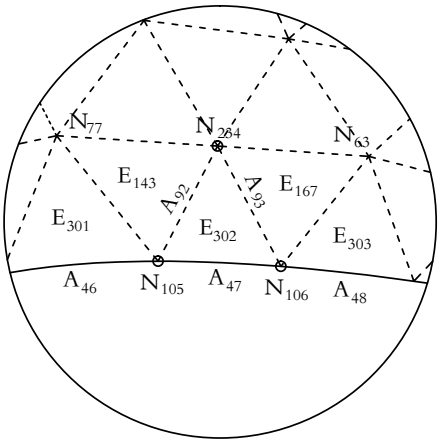
The nodes and edges on the upper part require no change except for the edges in full contact with the moving object (those edges that have two nodes on the moving body). Edges on the upper body no longer have the elements (triangles) and the nodes at their left sides after detachment and this is needed to be updated based on the defined data structure in the previous section (Fig. 3.17(b)). This step is crucial in order to make the procedure ready and compatible with a flow solver. This updating procedure carried on for the lower part is more

complicated than the upper part, as new nodes and edges are duplicated on this part of the moving object(Fig. 3.17(c)). Accordingly, those edges with two and one nodes on the body are needed to be updated. It is necessary to identify and extract all necessary information in the vicinity of the moving body before updating the data structure.



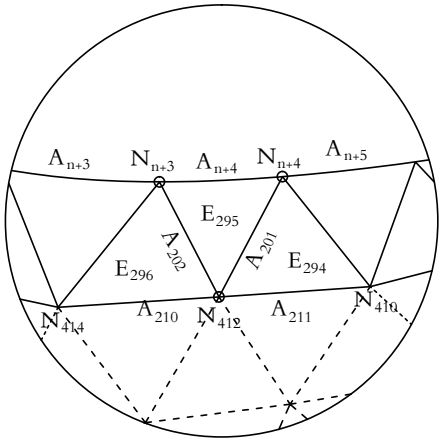
A_{47}	N_{105}	N_{106}	E_{302}	E_{295}	N_{234}	N_{412}	–
A_{92}	N_{105}	N_{234}	E_{143}	E_{302}	N_{77}	N_{106}	–
A_{93}	N_{106}	N_{234}	E_{302}	E_{167}	N_{105}	N_{63}	–
A_{201}	N_{106}	N_{412}	E_{294}	E_{295}	N_{410}	N_{105}	–
A_{202}	N_{412}	N_{105}	E_{296}	E_{295}	N_{414}	N_{106}	–

(a) edge arrangement before inflation



A_{47}	N_{105}	N_{106}	E_{302}	–	N_{234}	–	BC
A_{92}	N_{105}	N_{234}	E_{143}	E_{302}	N_{77}	N_{106}	–
A_{93}	N_{106}	N_{234}	E_{302}	E_{167}	N_{105}	N_{63}	–

(b) edge arrangement on the upper part of the moving object



A_{n+4}	N_{n+3}	N_{n+4}	–	E_{295}	–	N_{412}	BC
A_{201}	N_{n+4}	N_{412}	E_{294}	E_{295}	N_{410}	N_{n+3}	–
A_{202}	N_{412}	N_{n+3}	E_{296}	E_{295}	N_{414}	N_{n+4}	–

(c) edge arrangement on the lower part of the moving object

Figure 3.17 Data structure update

The overall algorithm for proposed procedure is given in Fig. 3.18.

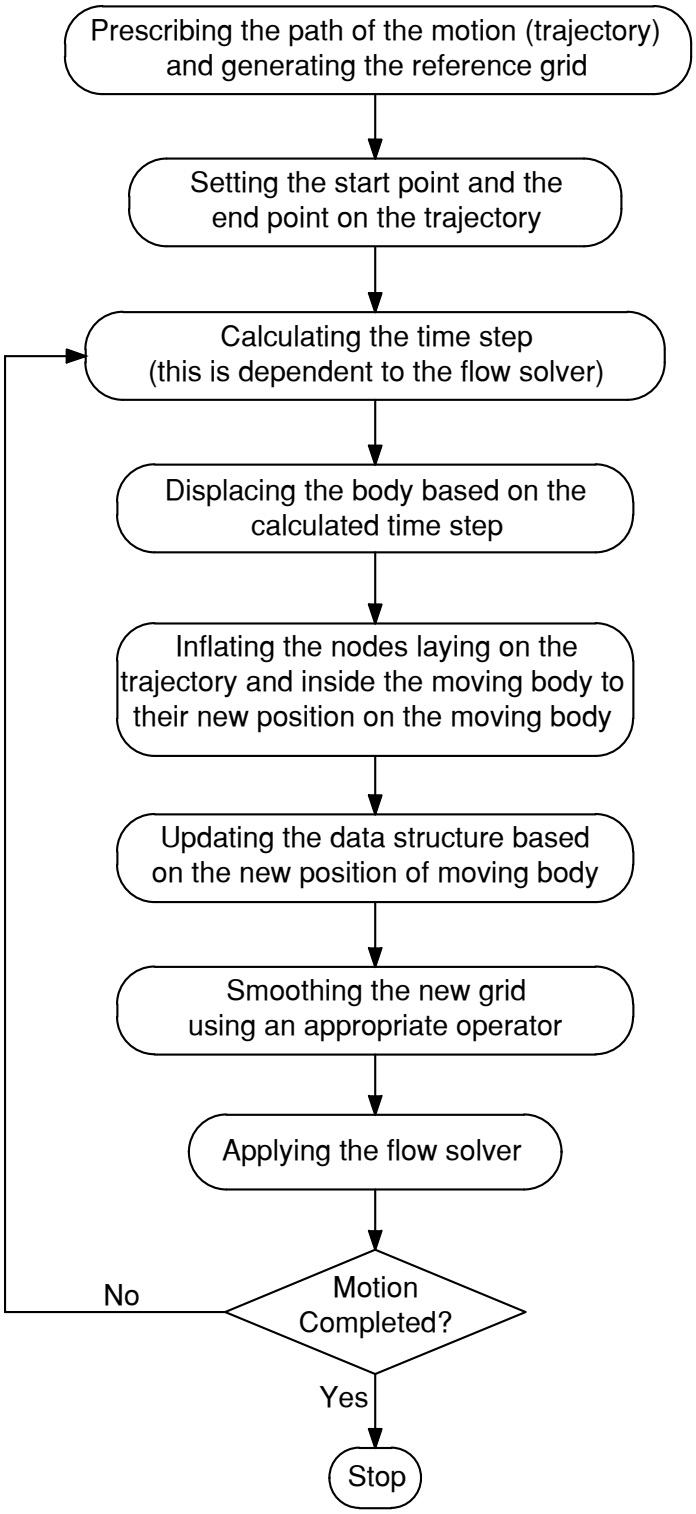


Figure 3.18 Proposed algorithm for translation grid motion

Fig. 3.19 illustrates the application of this procedure to the case of a circle moving past a series of bumps. This shows the evolving grid at four times steps. One can observe the position of two grid cells, #301 and #501, ahead of the body at step t_n, moving past the body on the upper and lower sides, respectively, at steps t_{n+m} and t_{n+2m}, and finally downstream at step t_{n+3m}. It is noted that the connectivity remains constant throughout the entire motion.

3.6 Rotation

3.6.1 Rotation Managed in Computational Space

The sliding procedure for rotary motions implemented in (Arabi et al., 2014) is obtained through the generic configuration of mapping an arbitrary body to a circle in computational space, which is rotated about its center in computational space. This is followed by the generation of a reference mesh in computational space which is then mapped to physical space. The result of this procedure is illustrated in Fig. 3.21 for a typical example.

The rotation is carried out in computational space by sliding a circle inside the body, which in computational space is also a circle, ie. the mapped body slides inside the mesh in computational space. The topological connection between the mesh nodes and the body nodes are modified for each angular position, followed by the solution of the mapping operator with the updated boundary conditions. The grid in computational space is fixed, only the connection to the boundary nodes change. The time steps are controlled in both, computational and physical spaces, as well as the angular rotation with respect to the motion defined in physical space. The pointers of the cells lying on the body are modified, and, as these refer to the physical boundary coordinates, they constitute a new set of boundary conditions for the mapping operators. These equations are solved in computational space at each step with the evolving boundary conditions in physical space. This procedure is repeated until the motion is completed in physical space.

This is shown in more detail in Fig. 3.22 for an example where for instance, for the element #350 (at the top of the diagram) the boundary pointers change from $62-61$ to $61-60$ and finally to $60-59$, for three consecutive steps. It is noted that this arrangement is mapped to physical space, and only the coordinates of the nodes change, not the connectivity.

A consequence of carrying out the rotation procedure in computational space, followed by a mapping of the grid to physical space, is that the mesh is constructed based on the geometry in computational space. As such, it may not be well-adapted with the real geometry in physical space, and the mesh thus generated may not have the required space-filling

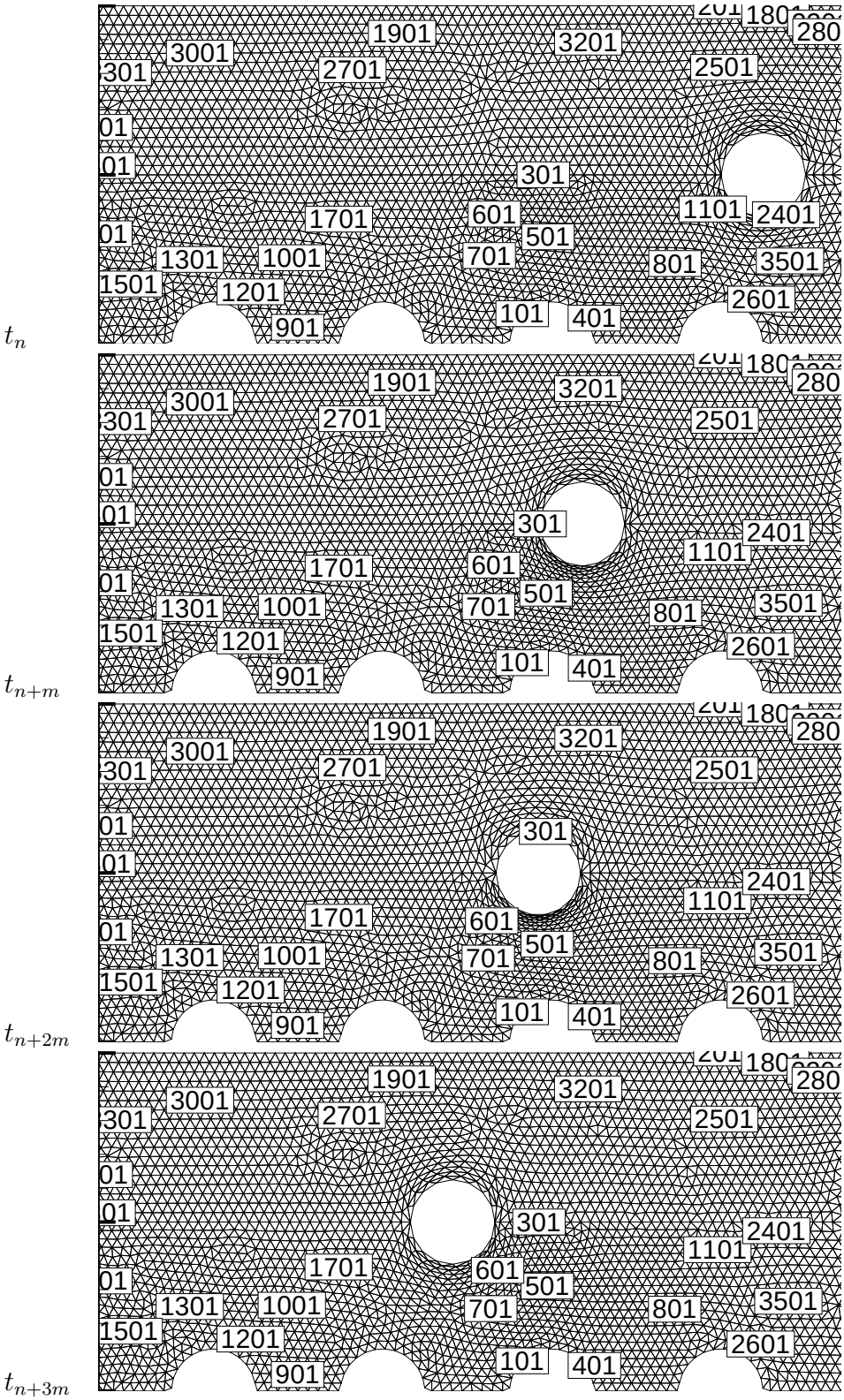
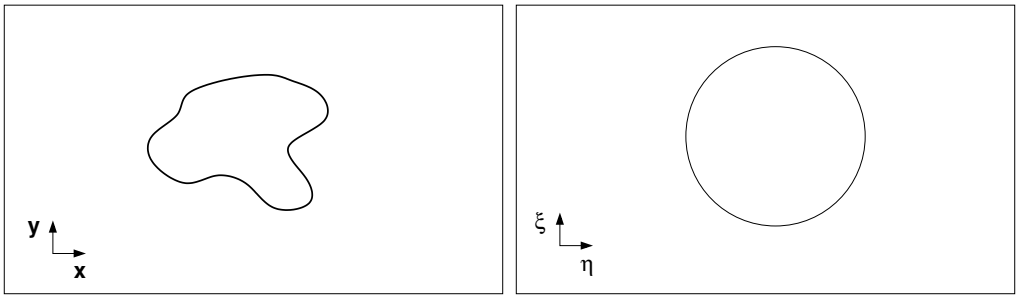
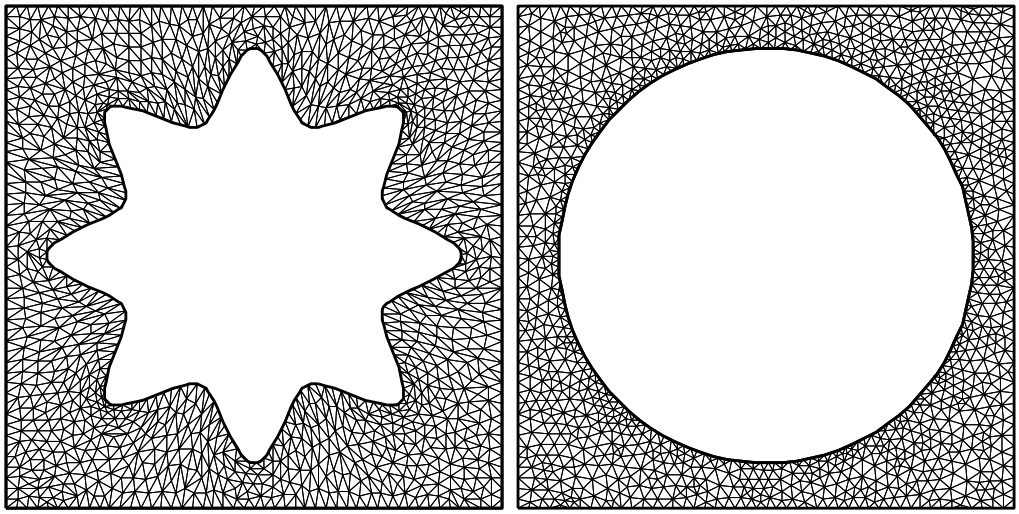


Figure 3.19 A moving circle passing several bumps



(a) Configuration in physical space (b) Generated configuration in computational space

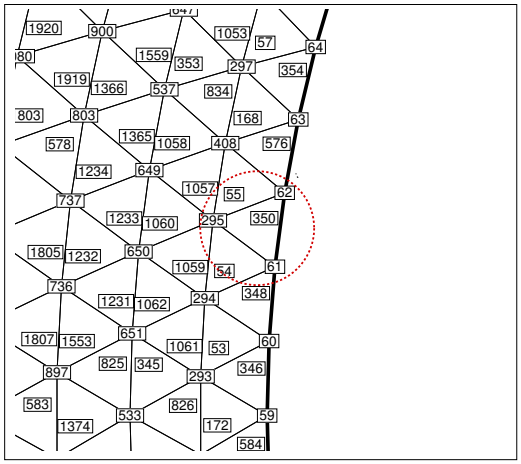
Figure 3.20 Generic mapping for rotational motion



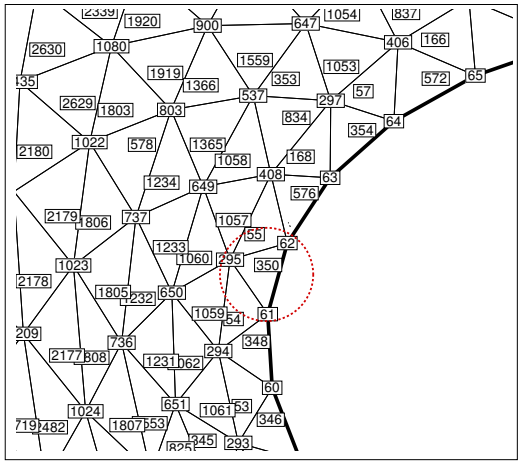
(a) Mapped mesh in physical space (b) Generated mesh in computational space

Figure 3.21 Mesh in physical space mapped from generic mesh in computational space

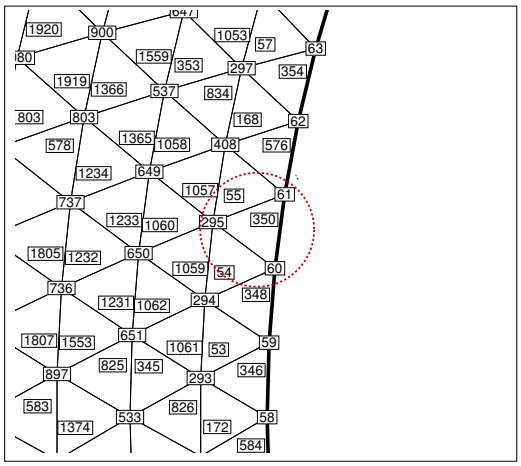
characteristics as in physical space. In addition, managing the motion in computational space and then mapping the grid to the physical space with the use of an operator (e.g. Winslow equation) always requires a high number of iterations since in each time step, the grid in physical space does not have any information from previous time steps. In other words, the method does not keep any track of nodes from the previous time step to be used as an initial guess for the next time step and always uses the same grid in computational space as an initial guess. This is one of the major drawbacks of the method which led to the modification for rotation in the present approach.



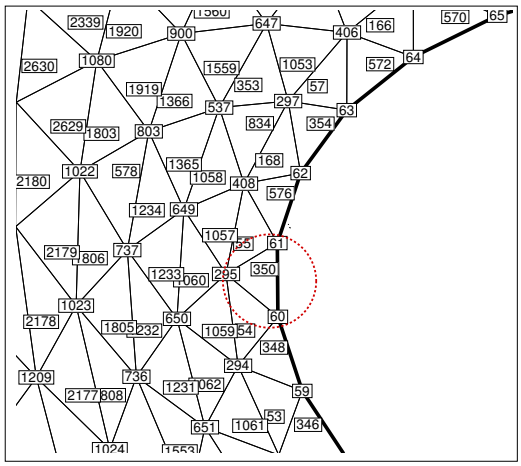
(a) Computational space, $t1$



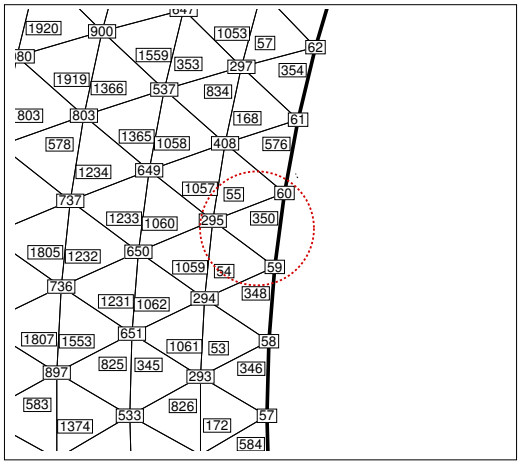
(b) Physical space, $t1$



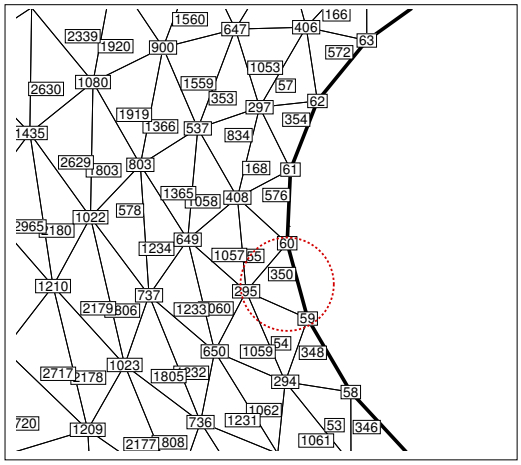
(c) Computational space, $t2$



(d) Physical space, $t2$



(e) Computational space,$t3$



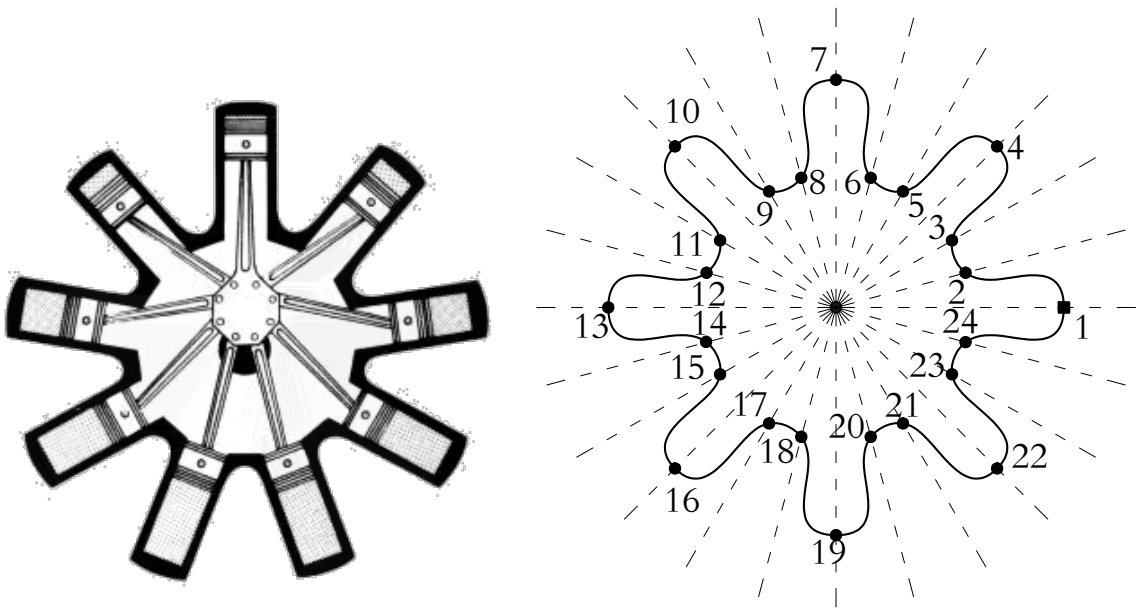
(f) Physical space, $t3$

Figure 3.22 Node numbering for cells connected to the rotating boundary in computational space and mapping to physical space for three consecutive time steps (Arabi-Narehei, 2012)

3.6.2 Rotation carried out in physical space

Similarly to the translation motion, in the current work, it is proposed to manage rotation in physical space. As concluded in the previous section, this approach yields improved space-filling characteristics as well as computational efficiency. Working directly in physical space requires fewer iterations for the smoothing procedure since the initial guess is directly available from the previous time step and is a good approximation of the updated grid.

To illustrate this new approach, the following analogy is proposed. The nodes on a rotational object are considered as pistons in a radial engine, as shown in Fig. 3.23. This means that the angular positions of these nodes are always fixed and they only have radial displacement in each time step (Fig. 3.25).



(a) Radial displacement of the pistons in a rotary engine

(b) Radial configuration of rotating body

Figure 3.23 Generic configuration for rotational motion in physical space

The displacement direction for each node is determined by connecting the node on the rotating object to the rotation axis, as shown in Fig. 3.24. The dashed line drawn from the rotation axis to the node in Fig. 3.24(a), defines the unit vector of this imaginary line which is constant during the rotation. This unit vector determines the line that intersects the rotated object and then gives the new position of the support nodes located on the rotated object.

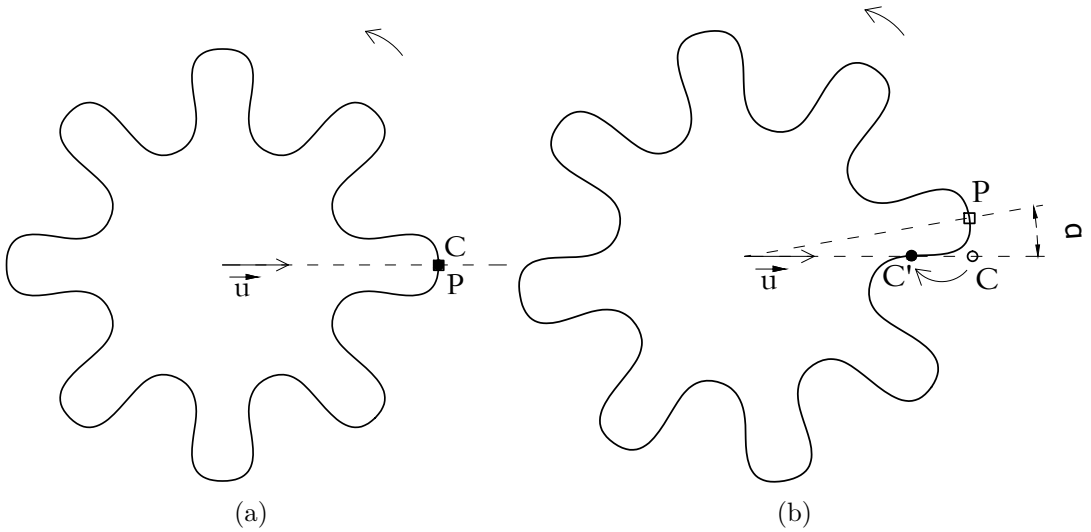


Figure 3.24 Node Projection at the intersection of rotating body and the line along the fixed unit vector u

Fig. 3.25 shows an object in four different angular positions. While a given point P corresponding to the object rotates, for the grid node #3, the angle of the line remains fixed but the radius changes at each time step. A step-by-step algorithm for this approach is given in Fig. 3.26.

While the idea of using projections simplifies the rotating procedure, for a case where the discretization of the rotating object is not homogeneous and the node concentration on the body must be maintained, the method is no longer useful as it cannot maintain the node distribution on the object. For instance, as shown in Figs. 3.27(a)(b), the node distribution is not homogeneous but it is needed to stay fixed during the rotation. In such a case, the object is rotated with the node distribution on it (Fig. 3.27(c)) and the nodes are renumbered since rotation changes the node ordering (Fig. 3.27(d)). In other words, nodes on the body shall be renumbered in order to respect grid connectivity and to prevent grid skewness.

Another remedy in order to have arbitrary node distribution and more control on the node concentration around the rotating object(s), is applying the method in multiblock grids, where a grid is rotating inside another one. This is presented in detail in the next chapter.

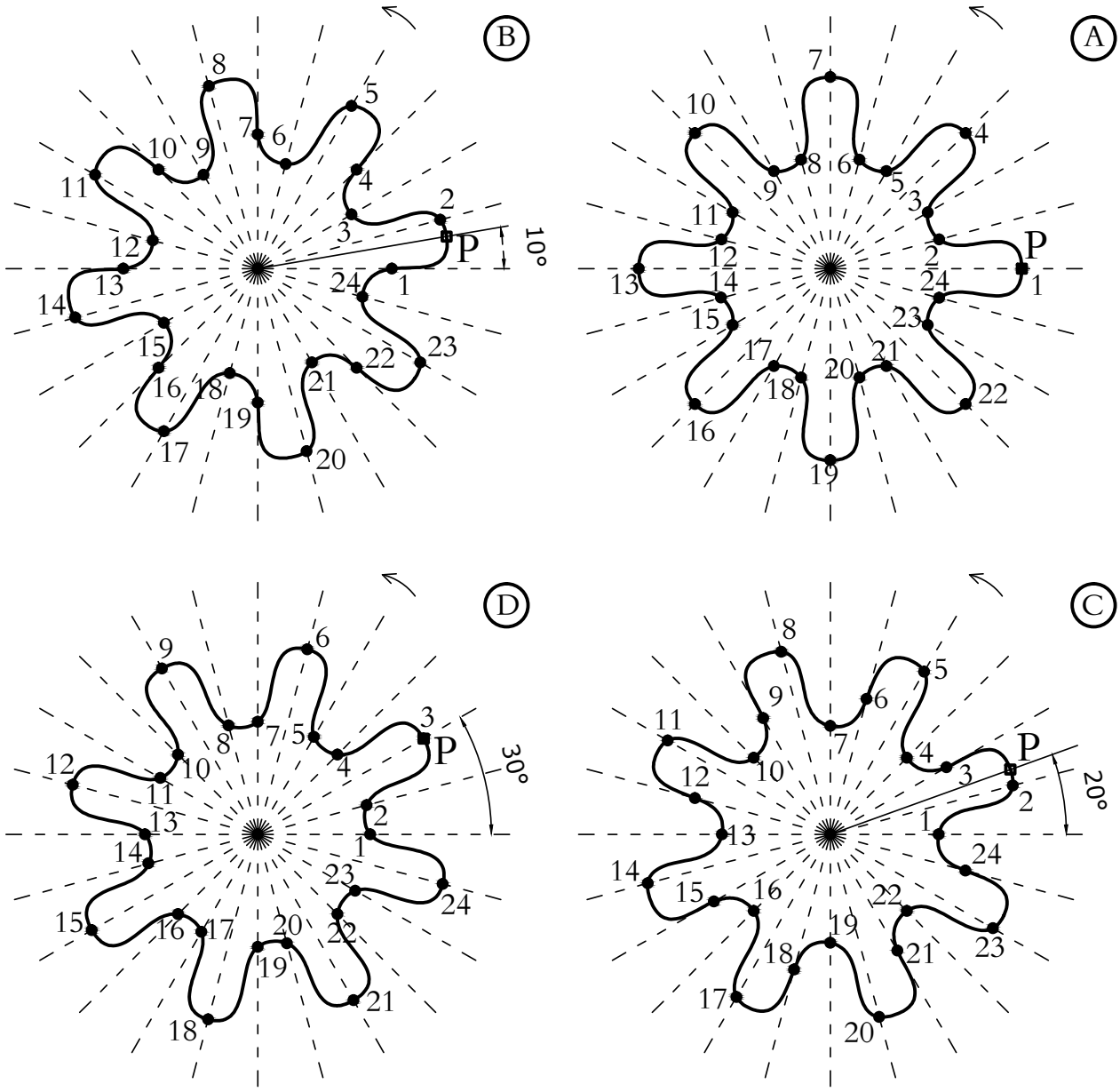


Figure 3.25 The angular positions of support nodes do not change during rotation.

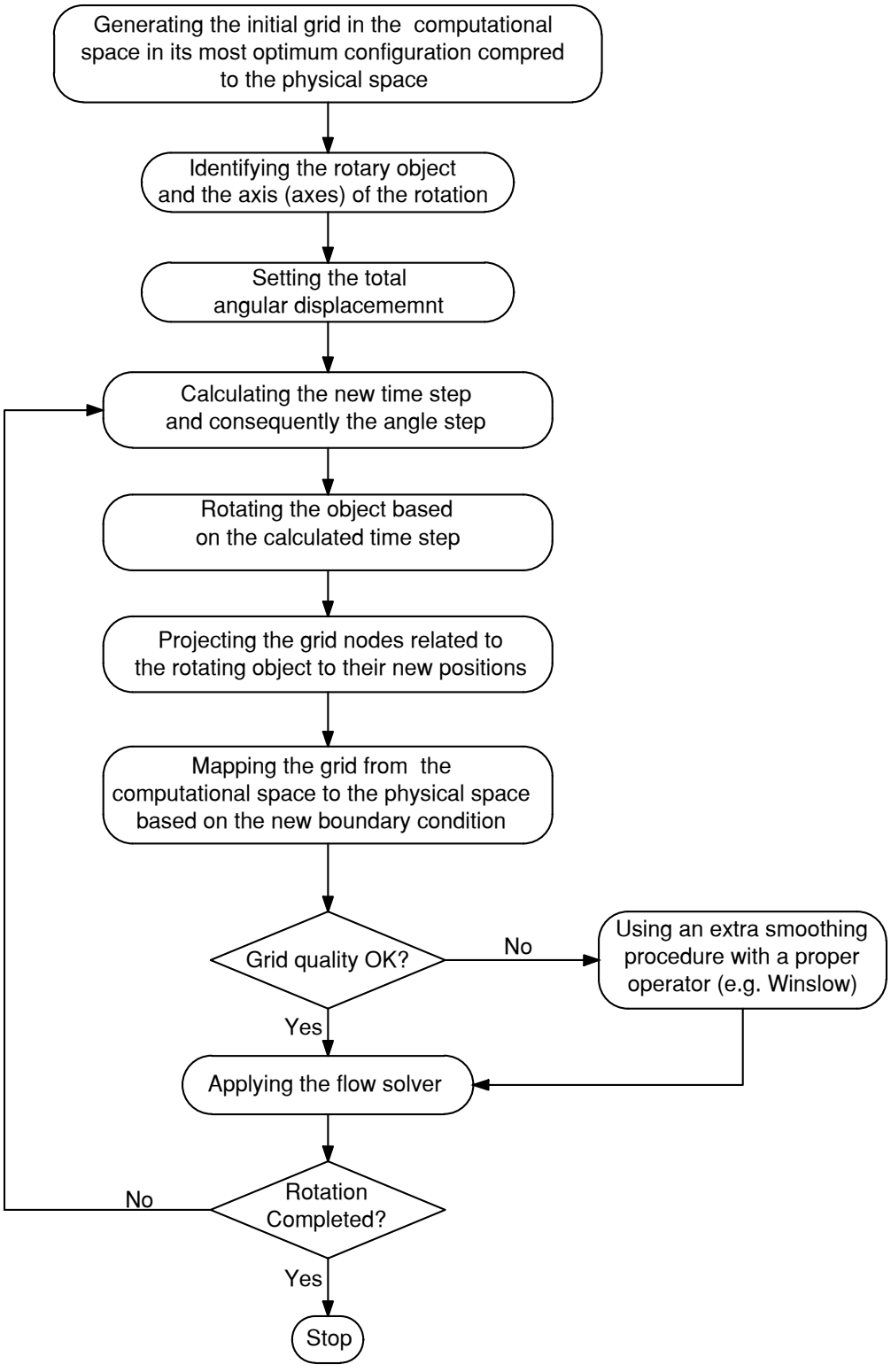
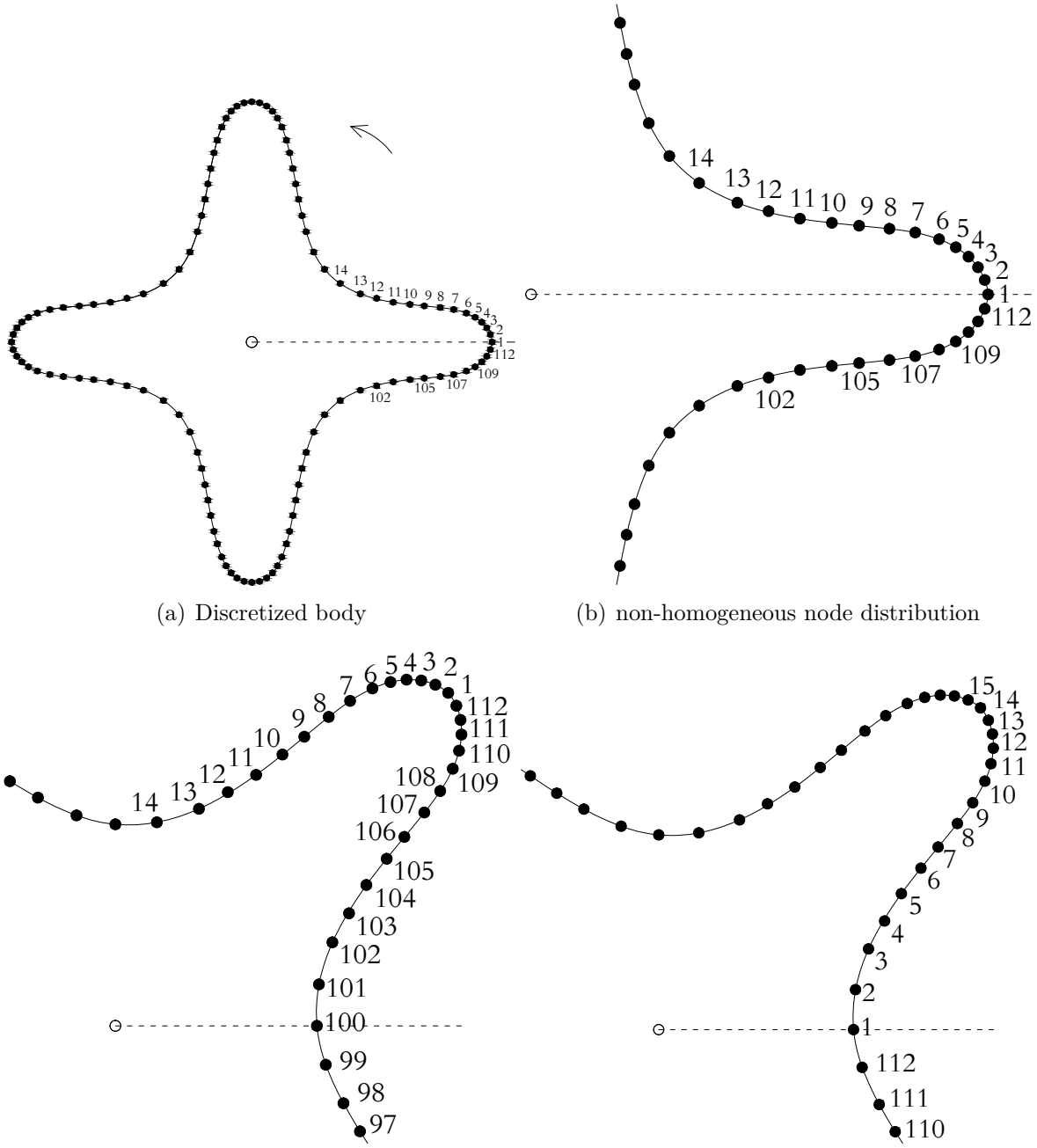


Figure 3.26 Proposed algorithm for rotational grid motion



(a) Discretized body

(b) non-homogeneous node distribution

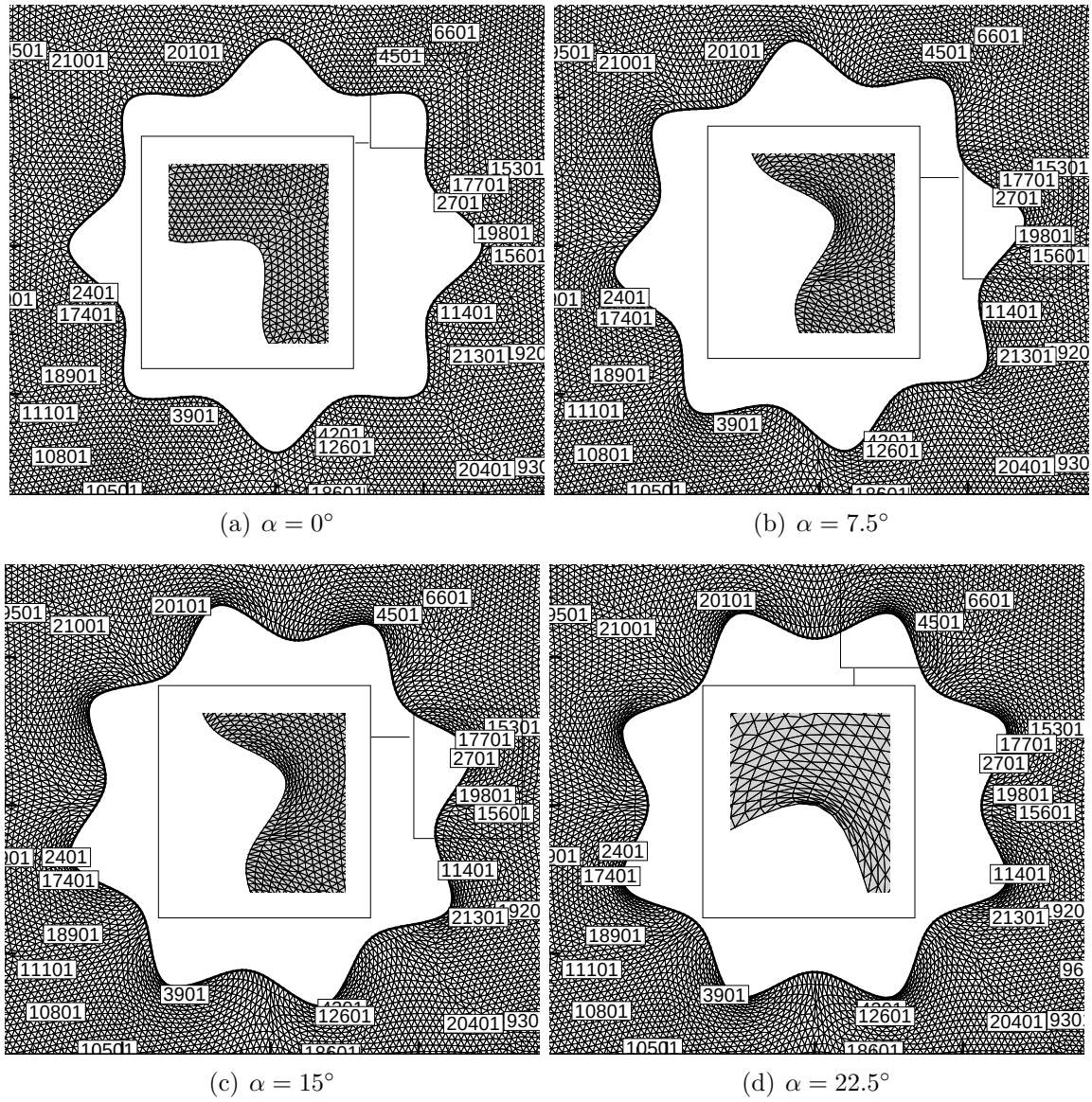
(c) Node distribution after rotation

(d) Node renumbering

Figure 3.27 Node adjustment to keep the node distribution on the object and to retain node ordering

3.6.3 Grid Quality Enhancement

The method for the rotation introduced in the previous section delivers effective results for rotating objects with small to moderate curvatures. Figure 3.28 illustrates the rotation of a boundary through an angle of 22.5° in four steps. One can follow the movement of cell #2401 (left most part of the figures) remaining attached to the surface as the wavy shape of the body moves past. Similarly, cell #20101, inside the mesh, is deformed and displaced by the rotation of the bump (upper part of the figures).



(a) $\alpha = 0°$ (b) $\alpha = 7.5°$

(c) $\alpha = 15°$ (d) $\alpha = 22.5°$

Figure 3.28 Grid rotation directly managed in physical space

However, the method cannot insure the reliability of the final grid and fails to handle the grid deformation for configurations with sharp curvatures. Figure 3.29 shows an example for the same basic shape but with increased amplitude of the waves of the boundary. This gives rise to a sharper variation of curvature. This effect on the mesh can be seen by comparing the initial mesh at the starting position of the rotation to that after a rotation of 20° shown in Fig. 3.30.

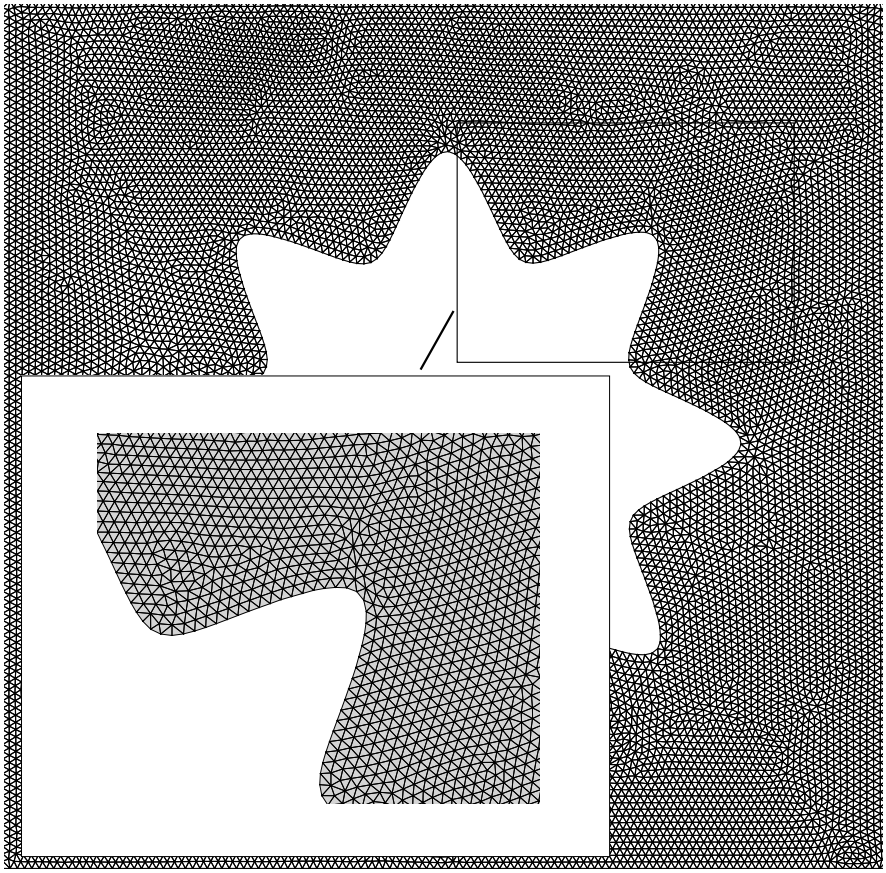


Figure 3.29 A configuration with sharp curvatures : initial mesh

The mesh is gradually deformed and the quality is highly affected around the bumps where sharp curvature is present. Two distinct behaviors are present in this example. First, the concentration of the cells at the top of the curvature peaks, shown in the zoomed portion at the upper left of the figure. Second, the more unwanted effect appears at the trough, between two peaks, where the cells suffer at large amount of stechtching, as can be seen at the lower left part in the zoomed part of Fig. 3.30.

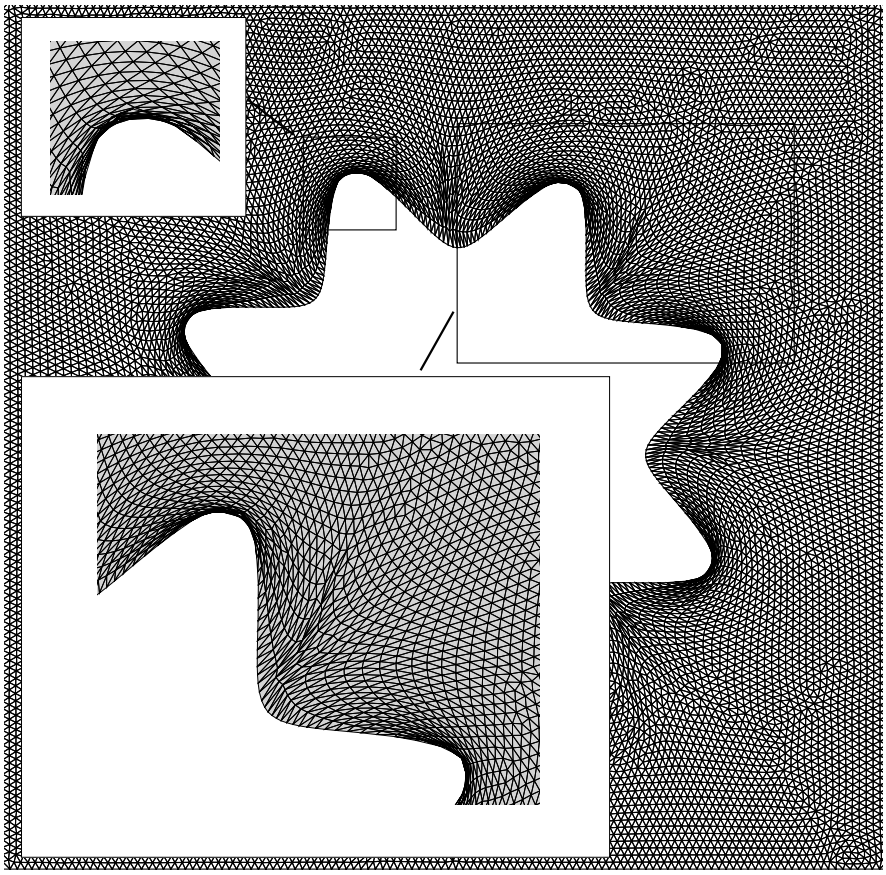


Figure 3.30 A configuration with sharp curvatures : mesh after rotation

Two issues arise in this situation. First is the computing cost related to a poor convergence of the iterative procedure. This is because of the fact that at each time step, the initial guess for the displacement of the nodes (i.e. the smoothing process) comes from the previous time step. The other issue relates to the nature of the model used to displace/move the nodes which will be discussed in Chapter 5. These are essentially smoothing procedures whose natural behavior is to pile up the elements in front of the moving boundary resulting in cells with poor quality after a few time steps. This is because the smoothing operator cannot push the grid cells far ahead from the moving boundary. This is clearly illustrated by observing the evolution of the cell shapes around the tips and troughs of the body curve oscillations, before and after the rotation. This shows a relatively well distributed initial mesh (Fig. 3.29) becoming increasingly concentrated after rotation (upper left part of Fig. 3.30. However, the most unwanted effect is the high stretching which occurs in the trough shown in the zoomed lower left part of the same figure.

To remedy this weakness, using a Poisson equation with forcing functions could provide

a better control on the smoothing procedure. However, devising efficient forcing terms is costly and significantly decreases the level of automation of the rotation procedure since these functions can only be defined a posteriori. A more effective method to enhance the grid quality after each time step is to start with a "better" grid as the initial reference mesh. The idea for such an extension to the generic mapping configuration lies in the choice of the location and size of the radius of the circle in computational space. Such a mesh should distribute and fill the space with cells adapted to the prescribed motion. This is illustrated in Fig. 3.32, where three configurations with different circle radii in computational space are meshed and mapped to the same physical space. The radii correspond to the outer, middle and inner radii of the shape in physical space, shown in Figs. 3.32(a), 3.32(a) and 3.32(a), respectively. The effect of the choice of the generic configuration is clearly illustrated, and results in mesh concentration at the tips of convex curvature of the boundary geometry, with a maximum for the smallest radius. The opposite effect, mesh stretching for concave curvature, takes place with a maximum effect for the largest radius.

The "optimal" configuration corresponds to a generic circle which is an "average" between the outer and inner most envelops of the geometry, as it would minimize departures from the curvatures of the geometry, as illustrated in Fig. 3.31.

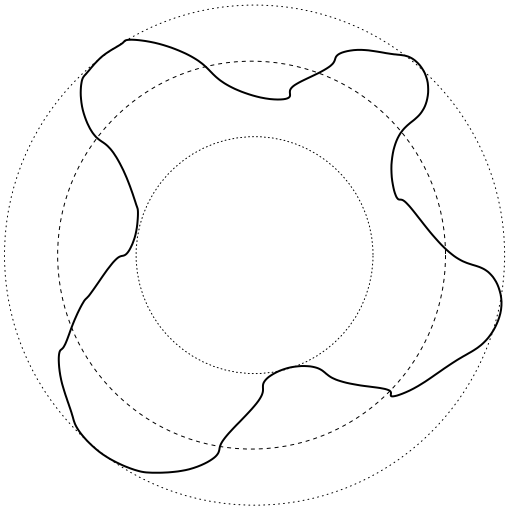
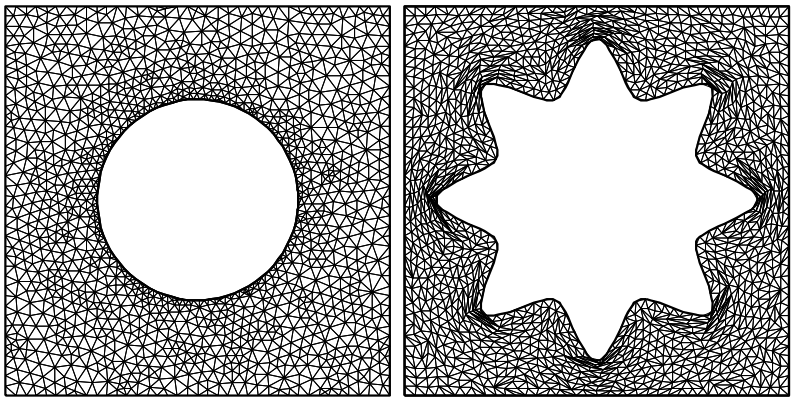
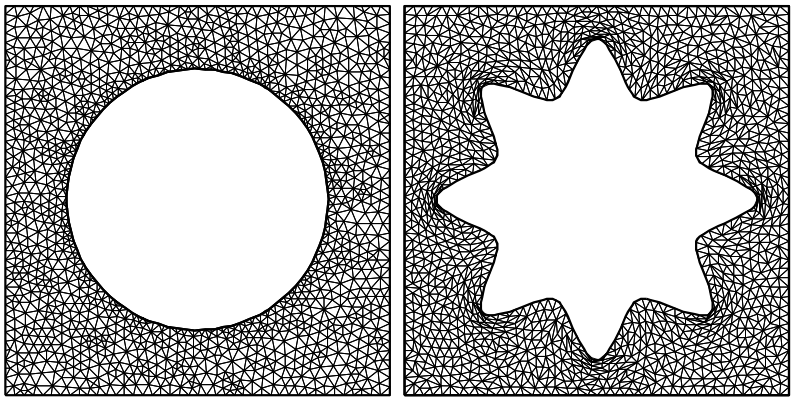


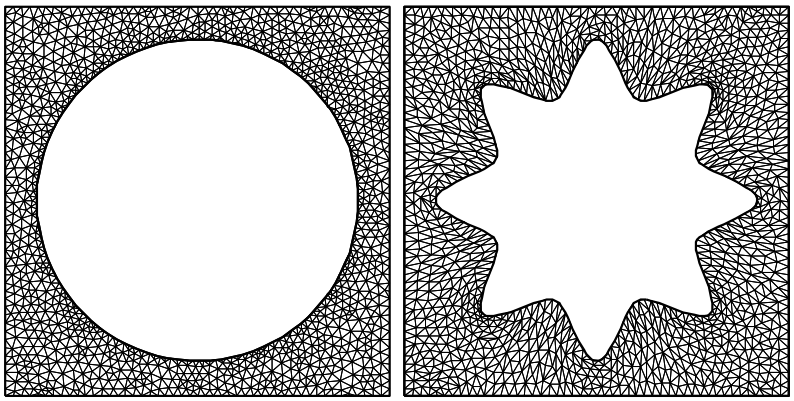
Figure 3.31 Optimal Generic configuration for complex body rotation



(a) Generic circle in computational space corresponding to innermost geometry



(b) Generic circle in computational space corresponding to middle geometry



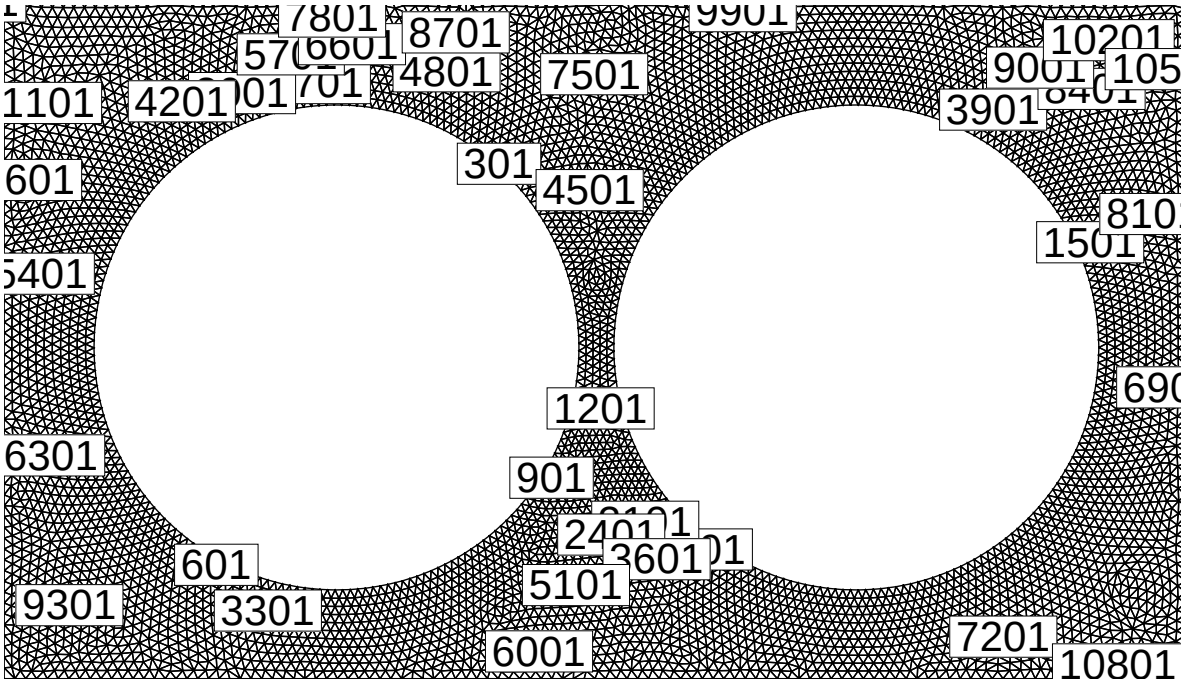
(c) Generic circle in computational space corresponding to outermost geometry

Figure 3.32 Three different generic meshes for the same geometry in physical space

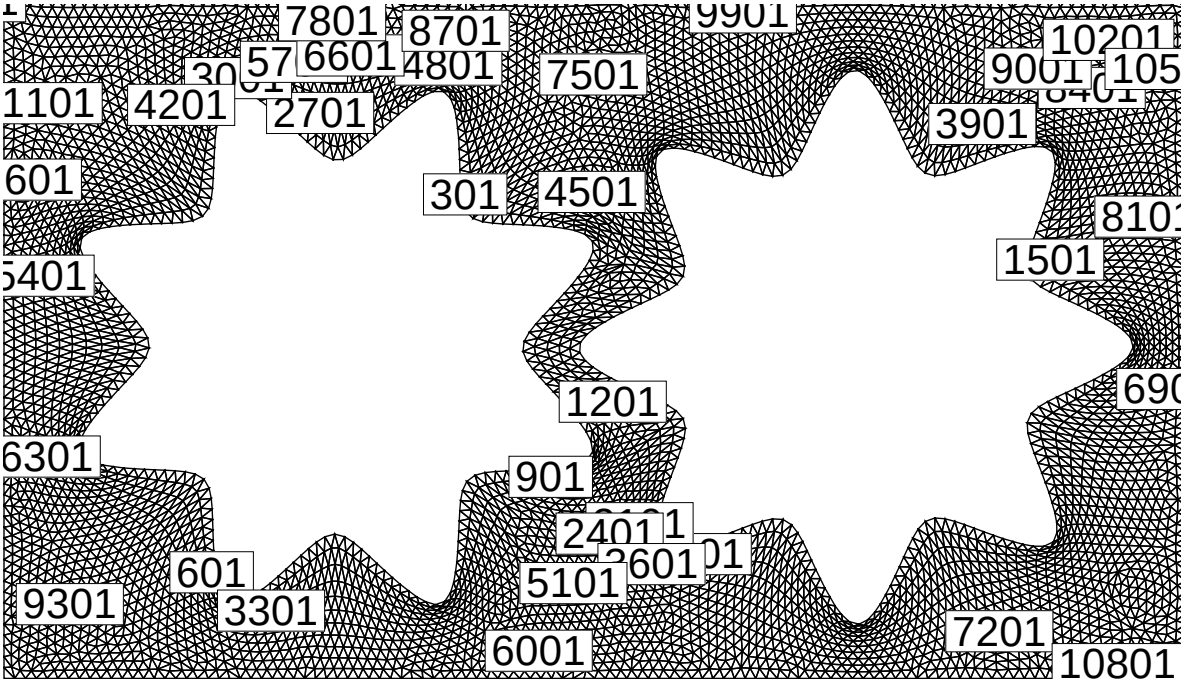
3.6.4 Multiple bodies in relative motion

The procedure for multiple rotary objects is quite similar to that with one single object, except that the rotational axes are different for each object and obviously the angular velocity for each object can be different.

Figure 3.33(b) shows two rotary objects in relative motion where the grid smoothing procedure uses one fixed grid (Fig. 3.33(a)) as the initial guess. In other words, in this figure with sharp curvatures, an operator maps a fix grid from the computational space to the physical space with the use of an initial reference grid instead of a grid in its last situation as an initial guess. A 22.5° rotation is shown for four different time steps in Fig. 3.34. As it can be seen from the figure, the connectivities remain fix and the grid is always valid. The full 360° rotation for this configuration can be watched on "`https://youtu.be/oNmxWfNFiyI`".

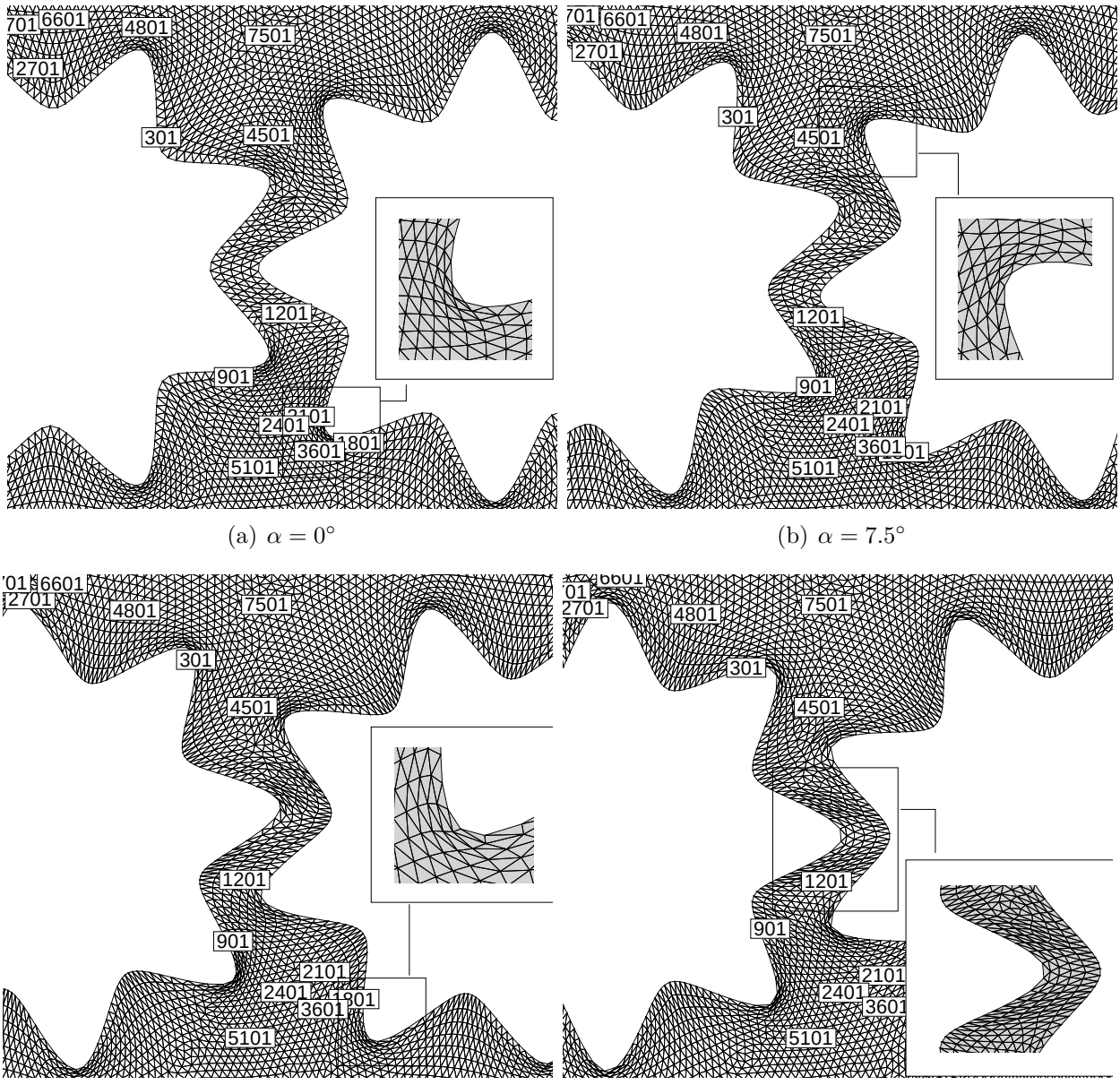


(a) Computational space



(b) Physical space

Figure 3.33 A configuration with two rotating object with sharp concave and convex curvatures



(a) $\alpha = 0°$ (b) $\alpha = 7.5°$

(c) $\alpha = 15°$ (d) $\alpha = 22.5°$

Figure 3.34 Relative rotation between two objects at four time steps

CHAPTER 4 APPLICATION TO HYBRID GRIDS

Hybrid grids are a combination of structured and unstructured grids used to handle grid generation for configurations that contain both regular and complex geometries. Some of the advantages of employing hybrid grids are :

1. Hybrid grids allow to control the orthogonality closed to the wall surfaces ;
2. The multiblock nature of hybrid grids allows to accelerate the convergence by dividing the calculation between several CPUs and also results in memory saving ;
3. The cell size of the grid can be easily controled in order to prevent unnecessary fine grids or to increase the grid density in critical areas.

In addition to these general advantages, extending the sliding procedure for hybrid grids is critical to account for the motion of bodies into regions of varying mesh density. In this chapter the proposed sliding mesh method is extended for hybrid grids in both translation and rotation. Finally, it is developed for motions which are a combination of translation and rotation.

4.1 Motivation

Hybrid grids are a particular class of composite grids whereby different types of meshes are generated in different zones to adapt to particular requirements, such as mesh density or the cell topology, i.e. structured or unstructured elements. For instance, in viscous dominated regions highly stretched cells to capture velocity gradients are appropriate, whereas isotropic cells would be used in the more uniform flow regions of the domain. This would give rise to zones with structured grids adjacent to zones with unstructured triangular elements. Another example is the transition of a region where the boundary is too complex to be discretized with structured cells, adjacent to an outer region where simpler structured or even cartesian grids would suffice. Such multiple blocks allow to resolve conflicting requirements, and facilitate the grid generation process by applying the appropriate grid characteristics (orthogonality, stretching, density...) selectively, where necessary. Admittedly, these goals can be achieved quite effectively by overset grids described in Section 2.3.3 whereby different types of overlapping meshes are generated to correspond to the characteristics of distinct regions. However, such approaches require the use of various complex and costly intergrid procedures.

In the case of moving grids, the transition of cells with specific characteristics in the vicinity of a moving boundary to those in the outer regions poses an additional difficulty.

This will be illustrated with the case of a body with a fine discretization, moving into a coarsely meshed region. This gives a reference or initial mesh consisting of uniformly coarse cells surrounding a finely meshed area around a body. As the body moves through the mesh, the coarse cells ahead of the body will reach and separate along the body surface where they will adapt to the fine discretization specified on the boundary. This yields very skewed cells. Meanwhile, the fine cells around the boundary will slide past the body, and will be shed into the downstream part of the domain. This will destroy the original reference mesh and the objective cell distribution. In fact, for the sliding mesh approach to yield the results shown in the previous chapter at Sections 3.2 and 3.6, it is implicitly required that the reference mesh be uniform. The ideal behaviour is that of a chimera grid, where a fine mesh attached to the body moves through a reference mesh without overlapping and the associated intergrid interpolation procedure. The sliding mesh approach can be extended to perform thus.

4.2 The Sliding Method for Hybrid Grids

Instead of having a body slide through a reference mesh, the proposed extension is to have a rigid mesh attached to a boundary slide through the reference mesh, as shown in Fig. 4.1.

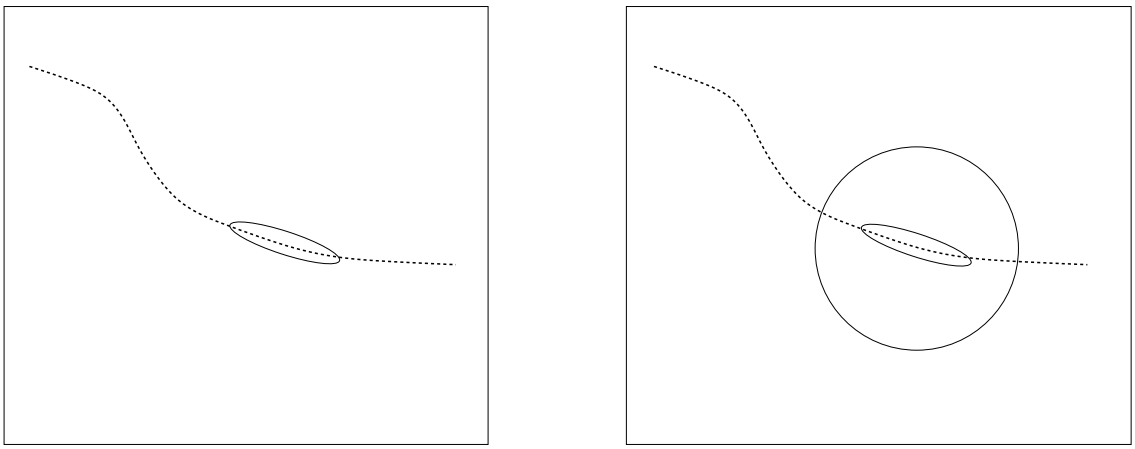


Figure 4.1 Sliding body extension to sliding mesh

The reference mesh is now composed of two contiguous grids separated by a permeable boundary (a circle in this example, but in general an arbitrary smooth closed curve). The internal grid, which is attached to the body, slides through the inside the external mesh, separating the cells on either side of the upper and lower parts of the trajectory. The permeable boundary now plays the role of the body. The overall procedure of the zipping method

extended for translation of a hybrid grid inside a main grid is as follows :

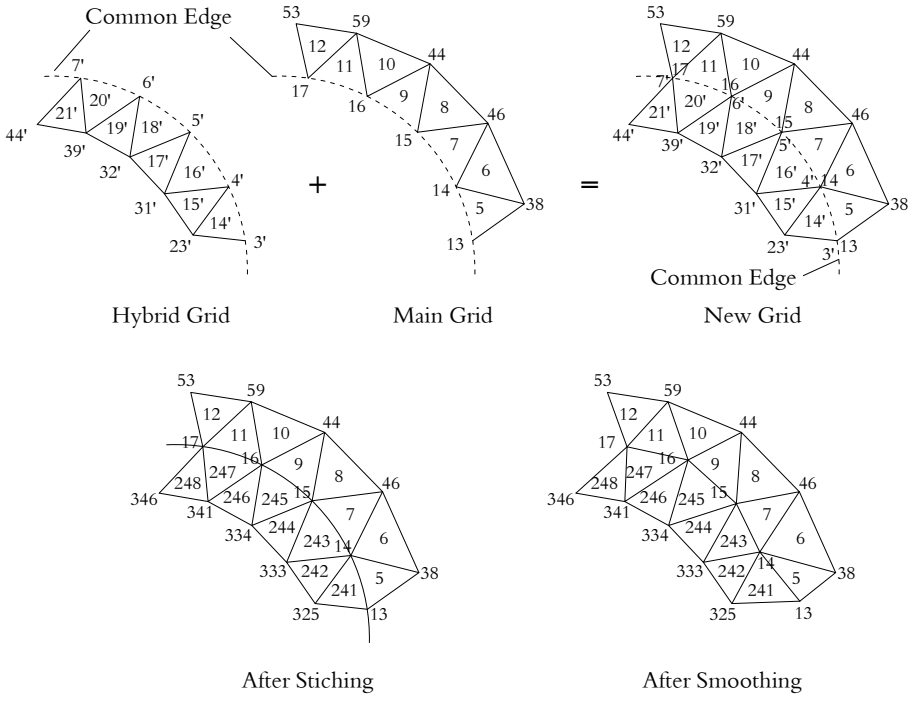
1. Prescribe the path of the motion (trajectory) and generate the initial reference grid ;
2. Select the most suitable configuration (generic shape) as the moving body inside the main grid. The hybrid grid will be fitted inside this generic shape later during the translation ;
3. Define the leading node and the trailing node on the generic shape. This is equivalent to the determination of upper and lower parts of the circle ;
4. Set the starting and end points on the trajectory ;
5. Determine the number of time steps (this is coupled with the flow slover time steps) ;
6. Displace the generic shape based on the time steps ;
7. Add the hybrid grid to the main grid (which will be fitted inside the generic shape) ;
8. Stitch the hybrid grid to the main grid ;
9. Reorder, renumber and correct element numbers, node numbers, side numbers and all other information based on the data structure ;
10. Smooth the new grid using an appropriate operator ;
11. Repeat the procedure from step 6) to the end point of the motion.

Extending the sliding approach for hybrid grids requires special grid management procedures since both grids (reference and body meshes) are to be stitched at the common permeable boundary. This is necessary for the smoothing procedure as well as the flow solver. According to the defined data structure in the previous chapter, identifiers for the topological entities (edges and nodes) on each side of common boundary need to be updated. This is followed with the removal of duplicated nodes and edges. Fig. 4.2 shows this renumbering procedure.

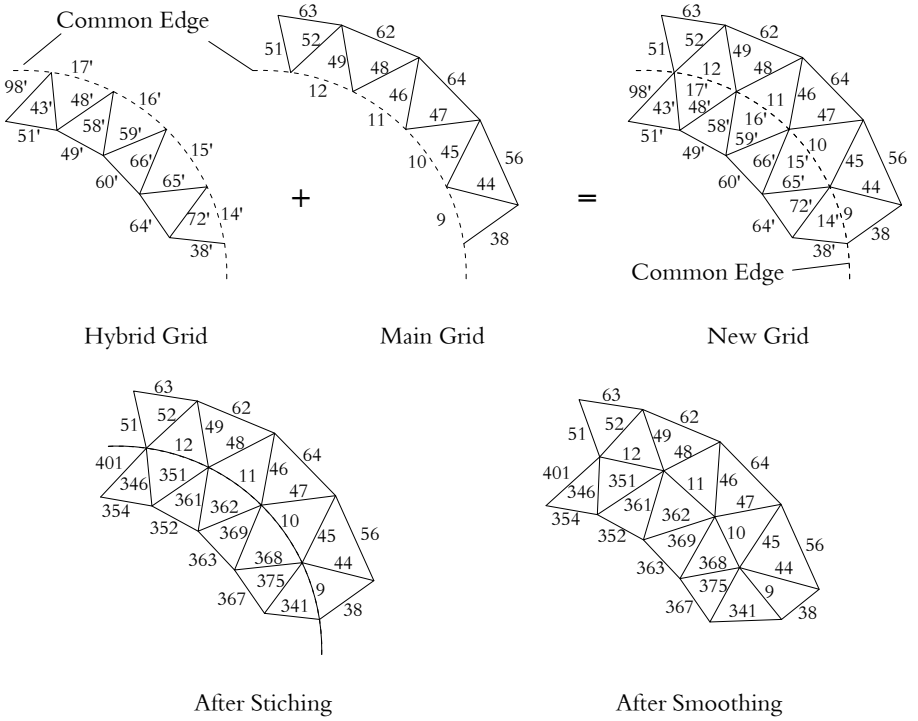
These procedures have been incorporated into the overall algorithm for various types of grid motion.

Translation

First, the simple translation of a body past a series of bumps is shown in Fig. 4.3 at several time steps. Refering to Fig. 4.1, one can see the division of grid into two grids ; the outer reference grid and the inner mesh attached to the moving body (an airfoil in this example). In the upper and lower "freestream" of the mesh, the path of cells #6001 and #151, respectively, slide past the inner mesh. It is noted that their position relative to the stationary bumps is constant. A similar path can be observed for cells #901 and #5701 whose



(a) Renumbering of nodes and elements



(b) Renumbering of edges

Figure 4.2 Renumbering procedure after stitching the reference grid to the body at each time step.

path is closer to the trajectory. Finally, cells #7351, #7501 and #7201 lie in the inner mesh, and their motion is clearly attached to that of the body. In addition, we have a dense grid around the airfoil and a medium size grid in other parts of the mesh.

This example clearly shows that the proposed extension has achieved the objective of a concentrated grid around a body which can move through a predefined reference grid along a given trajectory. In other words, controlling mesh concentration, mesh orthogonality and

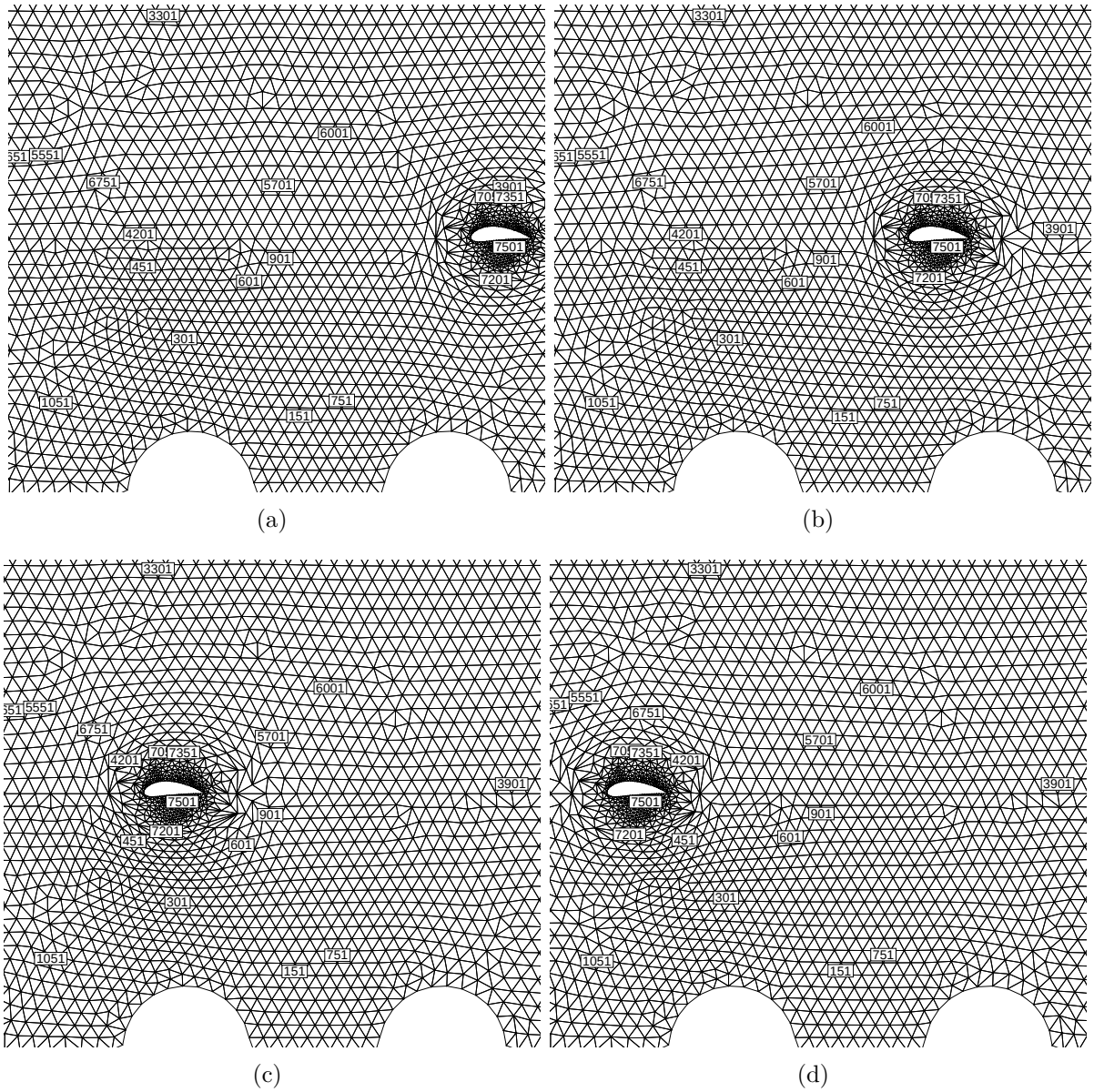


Figure 4.3 Grid motion induced by the translation of an airfoil at four different time steps.

other properties of the grid is now possible with the zipping method using composite grids. [1].

Rotation

Similar to the translation, extending the sliding approach to hybrid grids requires more arrangements since both grids (main grid and hybrid grid) must be stitched at the common interface. This is necessary for the smoothing procedure as well as the flow solver. According to the defined data structure in the previous chapter, values on each side of the interface must be updated at each time step. In addition, a generic configuration shall be considered as the rotating object body inside the main grid and the hybrid grid will be located inside this generic rotating object. In the current work, the rotating generic object is considered as a circle as it has no effect on the grid quality during the rotation. Like in the translation, the shape of the outer boundary of the hybrid grid shall be exactly the same as the rotating generic object inside the main grid, in order to be well-fitted inside the main grid. The procedure is as follows

1. Generate the initial grid in physical space including the generic rotating object ;
2. Identify the axis of the rotation which in this case is always the center of the circle ;
3. Add the hybrid grid to the main grid (which is fitted inside the generic shape) ;
4. Set the angle of rotation ;
5. Calculate the time step. This is coupled with the flow solver ;
6. Rotate the generic object around the rotation axis based on the calculated time step ;
7. Rotate the hybrid grid around the rotation axis based on the same time step ;
8. Stitch the hybrid grid to the main grid ;
9. Reorder, renumber and correct element numbers, node numbers, side numbers and all other information in accordance with the data structure ;
10. Smooth the mesh with the use of an appropriate smoothing operator ;
11. the process is repeated from step 6) to the end point of the motion to complete the entire rotation.

The stitching and renumbering procedure is quite similar to the translation which is shown in Fig. 4.2.

This procedure was applied to the simple rotation of a body in the wake of a stationary obstacle (in this example, both were chosen as airfoils) shown in Fig. 4.4. It can be observed that the grid concentration is implemented around the flap and the grid size is much smaller

1. The full motion can be watched on `https://youtu.be/QhEliuAo0r4`

in this area. Also, it can be seen that the nodes inside the hybrid grid are rotating with the flap (e.g. node number 4601) while the nodes outside this area are fixed (e.g. node number 201). [2].

Complex Arbitrary Motion

A final complex motion combining the translation of an airfoil rotating past a series of stationary bumps was computed and shown in Fig. 4.5. The described procedures for translation and rotation are applied simultaneously to handle this tumbling motion [3].

2. The full motion can be watched on `https://youtu.be/1HUCcp6EzN4`

3. The full motion can be watched on `https://youtu.be/QPOWlnaax_8` and `https://youtu.be/dY9gfo4i5_8`

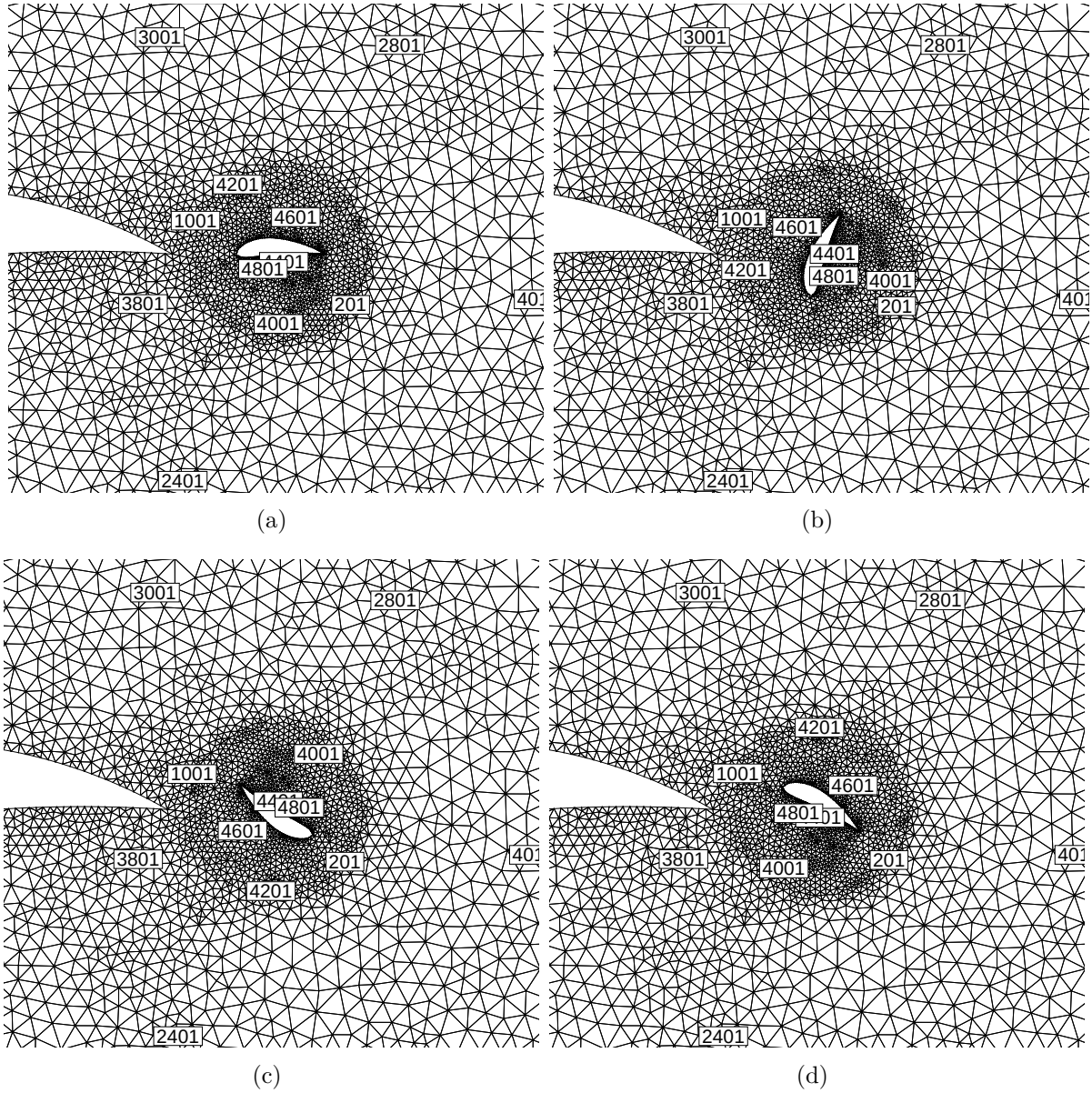
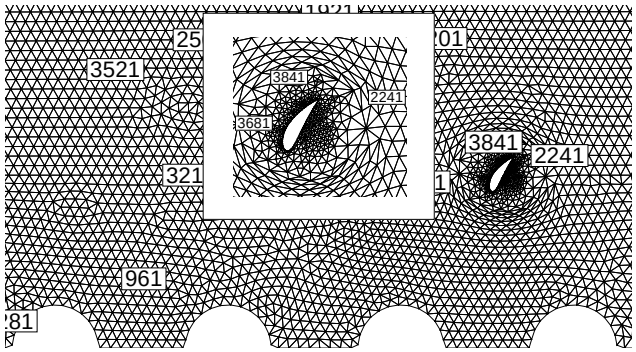
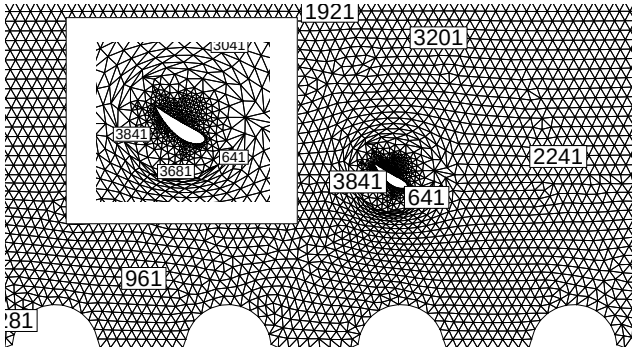


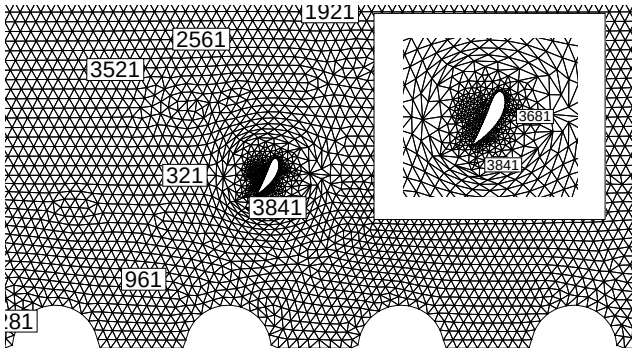
Figure 4.4 Grid motion induced by the rotation an airfoil at four different time steps.



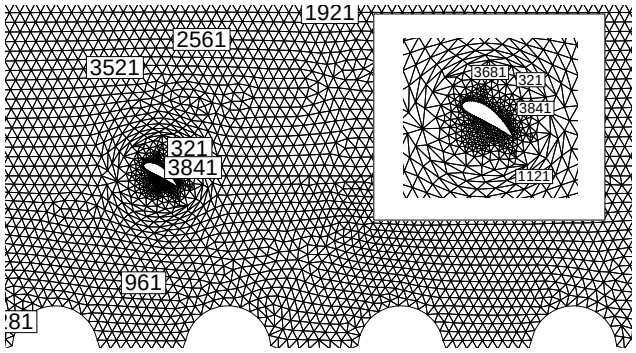
(a)



(b)



(c)



(d)

Figure 4.5 Combination of translation and rotation of an airfoil at four different time steps

CHAPTER 5 GRID SMOOTHING

Generally, grid smoothing is a post-processing procedure designed to improve the mesh quality. In the present study, it is the basis for moving or deforming the reference mesh subject to the location of the moving boundary nodes. As this procedure is applied at each time step, it is critical to devise an effective smoothing/moving operator for the moving procedure. To this end, two techniques based on the solution of elliptic equations are investigated. In the first instance, Laplace's equations are solved by a barycentric averaging procedure. A finite volume scheme based on an extension of Winslow's equation for unstructured grids is proposed. These are compared for computing efficiency and various grid quality criteria.

This chapter has been submitted as an article in the "International Journal of Fluid Machinery and Systems (IJFMS)" and was presented at the $27th$ IAHR Symposium on Hydraulic Machinery and Systems" (Falsafioon et al., 2014).

5.1 Unstructured Grid Smoothing

Grid smoothing can be carried out by various techniques where the nodal coordinates are modified as the solution of an operator such as Laplace or Winslow (Winslow, 1966), or the use of functionals (Castillo et al., 1988). These have been successfully used for structured meshes. However, elliptic smoothing has been challenging for unstructured grid generation due to the non-conservative form of these equations, as well as the lack of an implied unique computational domain. (Knupp, 1999) has shown unstructured Winslow mesh smoothing on unstructured quadrilateral meshes using a locally defined computational domain. To remove these restrictions concerning the application of elliptic smoothing methods on unstructured meshes, (Karman and Sahasrabudhe, 2007) and (Karman Jr, 2010) proposed a local mapping based on computational "virtual control volumes" where the neighbouring cells around a given node are placed on a unit circle in computational space and mapped to physical space. Recently, in a similar fashion, (Arabi et al., 2014) also addressed these two major problems. The non-conservative aspect of the formulation, was handled by introducing a 9-point finite difference stencil around each node of the mesh to correctly represent the mixed derivatives in Winslow's equations, instead of augmented cells. The mapping procedure was resolved by employing an explicit overall mapping. However, this approach is strongly dependent on the choice of the mapping configuration which can be arbitrary for a general unstructured mesh.

In the current study, a combination of the local mapping approach of (Karman Jr,

2010), along with an operator such as Laplace or Winslow is investigated. A comparison and assessment of these two techniques, based on two quality measures (shape factor and minimum angle which both are local criteria) is carried out. Finally, a global quality smoothness criterion is introduced and used to assess global smoothing characteristics of these methods.

5.2 Elliptic Smoothing

Laplacian smoothing is the simplest and consists in solving the following system of partial differential equations for (x, y),

$$\begin{aligned} \nabla^2 x &= 0 \\ \nabla^2 y &= 0 \end{aligned} \tag{5.1}$$

In this work, this will be carried out simply by using a barycentric averaging procedure of the coordinates connected to each node of the mesh.

$$\begin{aligned} x_i &= \frac{\sum_{j=1}^{n} x_j}{n} \\ y_i &= \frac{\sum_{j=1}^{n} y_j}{n} \end{aligned} \tag{5.2}$$

where n is the number of nodes $j = 1, n$ around the specific node i of the mesh. This yields smooth meshes for most regular geometries, but invalid meshes can result for configurations with concave corners when large changes of curvature occur.

The validity of this mapping can be extended by inverting the independent with dependent variables in Eqns. 5.1. Thus reformulated, this yields the Winslow equations (Winslow, 1966) in computational space (ξ, η), given as,

$$\begin{aligned} \mathcal{L}(x) = &\quad g_{11} x_{\xi\xi} - 2 g_{12} x_{\xi\eta} + g_{22} x_{\eta\eta} &= 0 \\ \mathcal{L}(y) = &\quad g_{11} y_{\xi\xi} - 2 g_{12} y_{\xi\eta} + g_{22} y_{\eta\eta} &= 0 \end{aligned} \tag{5.3}$$

where

$$\begin{aligned} g_{11} &= x_\eta^2 + y_\eta^2 \\ g_{12} &= x_\xi x_\eta + y_\xi y_\eta \\ g_{22} &= x_\xi^2 + y_\xi^2 \end{aligned}$$

In (Knupp and Steinberg, 1993) it is mentioned that while the Winslow operator guarantees continuous global mapping, truncation errors can lead to folded meshes. In such instances, additional control is needed to adapt the mesh around the boundaries to insure the validity of the results, especially around discontinuous parts of the physical boundary. A method based on Taylor series expansion to solve this operator on equilateral triangles where all the angles are equal to $\pi/3$ has been proposed by (Knupp, 1999), but is too restrictive for general applications. Another method has been proposed in (Karman Jr, 2010), based on generating a virtual control volume in the physical domain locally around each node, as a local computational space with the same number of neighbouring nodes as in physical space. (Arabi-Narehei, 2012) compared two techniques for the solution of the functional operator using finite volume and finite difference methods. The latter was based on inserting a 9-point cartesian stencil around the node (ξ_i, η_i) (Fig. 5.1).

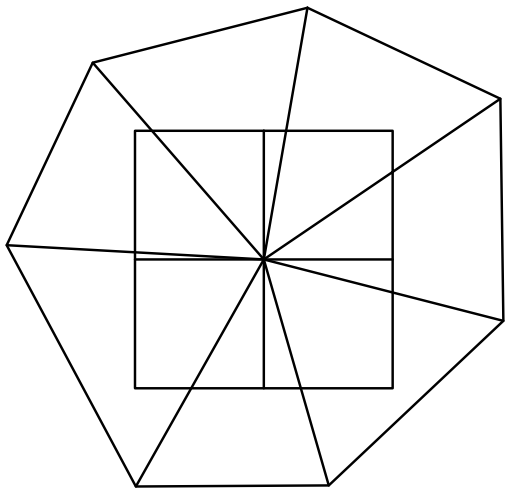


Figure 5.1 A 9-point Cartesian stencil around each node of the unstructured grid in computational space

The values on each point of the stencil can be calculated with an interpolation method. With these values, first and second order derivatives are approximated to second order accuracy on the cartesian stencil with an equal spacing ($\delta\xi_i = \delta\eta_i$) using a finite difference scheme. The method fails to generate a valid mesh for some particular situations, specially around sharp corners as it could give rise to an uneven contribution. In the current work, an extension is used whereby Winslow equations are solved using the local computational space method of (Karman Jr, 2010) combined with averaging method presented by (Arabi-Narehei, 2012) for the evaluation of cross-derivatives.

5.2.1 Physical and Computational Domain

A control volume is created in physical space by joining the centroid of each element connected to a node to the mid points of the element sides that pass through that node i. This creates a closed polygonal control volume with $2m$ sides (faces) ; where m is the number of elements surrounding node i. Each element contributes 1/3 of its area to the control volume area and the volumes from all the nodes cover the domain without overlap (Fig. 5.2).

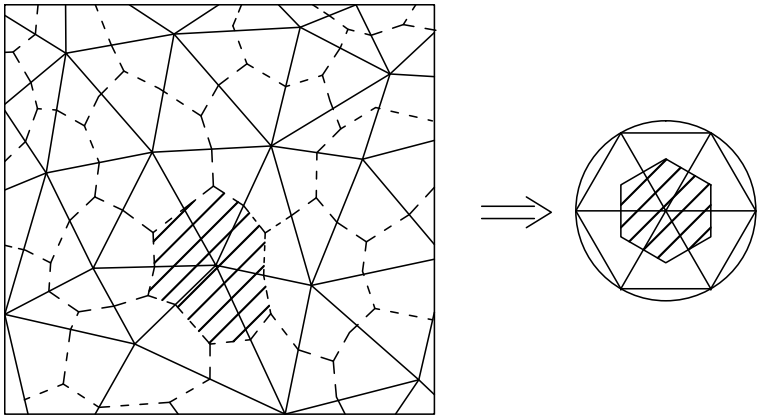


Figure 5.2 Local mapping from physical to computational space

The idea of virtual control volume manipulation is reintroduced with the goal of adapting an unstructured mesh to match a desired spacing. This is done in an iterative manner adapting virtual control volumes to achieve equal weights across stencil edges with the weights determined by a stationary scalar field. Elliptic equations enforce a smoothness condition between the spacing in the computational domain (ξ, η) and the physical (x, y) as a result of the min/max principle. The key to optimizing a mesh through the use of elliptic equations lies in using a regular computational domain. In structured meshes, the computational domain has the same topology as the physical space, and ideal spacing i.e $\delta x = \delta y$, thus effectively enforcing an equal spacing around each node. For an unstructured mesh, there is no such universal computational domain that matches every unstructured topology ; therefore, for each unstructured mesh a computational domain must be devised to match the topology of the physical mesh. A solution to this problem was proposed by (Sahasrabudhe, 2008) with the introduction of computational domains that are only defined locally. For a stencil of elements surrounding a single node, an ideal mesh can be defined by equally spacing the connected points on a unit circle. These stencils, called virtual control volumes, are locally defined and can be used to drive the solution of the smoothing equations. Thus formulated, smoothing by the use of elliptic operators is a distinct boundary value problem with dirichlet boundary

conditions for each node.

5.2.2 Finite Volume Scheme

The Winslow equations are in non-conservative form and the three coefficients, $g11, g12$ and $g22$ in Eqns. 5.3 are functions of the gradients of the dependent variables in the computational space. Using a linearization procedure, these terms can be integrated separately over a control volume defined around each point of the mesh in computational space.

The integration path for the application of Green's theorem is formed by joining the centroid of each triangular element to the midpoints of its sides, as shown by the dashed lines in Fig. 5.2. The cell edges divide each triangular element into three equal areas, and, collectively, these areas form non-overlapping contiguous control volumes associated with a node in the mesh. The hatched region in Fig. 5.2 indicates a control volume with a centroid node which is the storage location of all dependent variables. Linearizing Eqns. 5.3, results in the following integral form.

$$g_{11} \iint_{\Omega} x_{\xi\xi} d\Gamma - 2g_{12} \iint_{\Omega} x_{\xi\eta} d\Gamma + g_{22} \iint_{\Omega} x_{\eta\eta} d\Gamma = 0 \tag{5.4}$$

Where Ω is the control volume created around each node. Applying the divergence theorem to the second order derivative terms, $x_{\xi\xi}$ and $x_{\eta\eta}$, gives,

$$\iint_{\Omega} x_{\xi\xi} d\Gamma = \iint_{\Omega} \bigtriangledown \cdot F d\Gamma$$

where the components of function F are $(x_{\xi}, 0)$. Similarly, for the cross derivative terms, for instance $x_{\xi\eta}$ we have,

$$\iint_{\Omega} x_{\xi\eta} d\Gamma = \iint_{\Omega} \bigtriangledown \cdot Q d\Gamma$$

While mathematically $x_{\xi\eta} = x_{\eta\xi}$ for continuous functions, numerically this yields different results depending on the order of the integration. (Karman Jr, 2010) has proposed a rigorous procedure resulting in a complex scheme using equilateral cells. In (Arabi et al., 2014) it was found that splitting the cross derivatives in two components and applying Green's theorem to both and taking an average, avoids the generation of folded cells in the physical domain. To simplify the algorithm, the cross derivative term is computed in two manners. Using $Q_1 = (0, x_{\eta})$ and $Q_2 = (x_{\xi}, 0)$, and using Q as the average,

$$Q = \frac{1}{2}(Q_1 + Q_2)$$

Integrating over the control volume and applying the divergence theorem for each dependent

variable, for example ξ, gives,

$$\iint_{\Omega} \bigtriangledown \cdot F d\Gamma = \oint F.\hat{n} dS \tag{5.5}$$

The term on the RHS integral represents the net flux through the surface of the volume and, for Winslow's operator, can be evaluated as,

$$g_{11} \oint x_{\xi} n_x dS - 2g_{12}[\frac{1}{2}(\oint x_{\eta} n_x dS + \oint x_{\xi} n_y dS)] + g_{22} \oint x_{\eta} n_y dS = 0 \tag{5.6}$$

$$g_{11} \oint y_{\xi} n_x dS - 2g_{12}[\frac{1}{2}(\oint y_{\eta} n_x dS + \oint y_{\xi} n_y dS)] + g_{22} \oint y_{\eta} n_y dS = 0 \tag{5.7}$$

where in counter-clockwise direction, the lengths of the sides of each control volume are calculated by,

$$\begin{aligned} n_x dS &= \Delta\eta \\ n_y dS &= -\Delta\xi \end{aligned}$$

(Arabi et al., 2014) concluded that for the cross-derivative terms, applying Green's theorem only for one component on each control volume side around node (ξ_i, η_i) yields a degenerated final mesh in most cases of geometries with severe boundary curvature variations. In other words, for arbitrary deformations in the (x, y) plane, the values of the calculated fluxes in (ξ_i, η_i) are dominated by the values from the cross derivatives terms. Moreover, taking only one component of the cross derivative term after applying the Green's theorem, deflects the final mesh in one direction. In (Karman Jr, 2010) these terms were calculated using a set of augmented cells attached to the control volumes. Therefore, the shape of virtual control volumes are different for the cross-derivative terms. The averaging procedure maintains the symmetry of the final mesh when the deformation of the physical boundary is symmetric.

5.3 The Discrete Equation

For each element, nodes are labelled 1, 2 and 3,in a counter-clockwise direction. Values of the dependent variable x and y are calculated and stored at these nodal points. In this way, a quantity ϕ representing x or y, at an arbitrary point within the element can be interpolated linearly,

$$\phi \approx a\xi + b\eta + c, \tag{5.8}$$

where ϕ represents x or y, and coefficients a, b and c satisfy the nodal relationships

$$\phi_i = a\xi_i + b\eta_i + c, \qquad i = 1, 2, 3 \quad \text{and} \quad \phi = x, y \tag{5.9}$$

such that, over the element, the continuous unknown field can be expressed as the linear combination of the values at nodes $i = 1, 2, 3$

$$\phi(\xi, \eta) \approx \sum_{i=1}^{3} N_i(\xi, \eta) \tag{5.10}$$

for ϕ representing x or y, and where N_1, N_2 and N_3 are the shape factors of each element and are respectively a, b and c in Eq. 5.9.

The main step is formulating the continuous description of the governing equations into an algebraic discrete form. Eqs 5.4 are the integral form of the governing equations. As indicated in Fig. 5.2, the domain in the integral form can be any arbitrary closed area in domain Ω, including areas that share common surfaces with the boundary Γ of Ω. Converting Eqs. 5.4 into a set of discrete algebraic equations in terms of the nodes distributed thoughout Ω is as follows :

1. The control volumes and all the neighboring nodes and edges attached to each node i are identified ; this requires an appropriate data structure (see Chapter 3) ;
2. Using numerical integration and the shape function approximations for the values (x and y) in the i_{th} element, neighbor of the node, Eq. 5.4 is expanded in terms of the nodal values of x and y.
3. On gathering terms, the resulting equation for node i can be written in the general discrete form

   $$a_i \phi_i = \sum_{nb} a_{nb} \phi_{nb} \tag{5.11}$$

   for ϕ representing x or y,and where a_i and a_{nb} are coefficients for the unknown nodal values of ϕ. This equation provides an algebraic relationship between the value of ϕ at the node i and the neighboring nodes in its vicinity.

$$\phi_i = \frac{\sum_{j=1}^{n} \phi_j}{n} \tag{5.12}$$

5.4 Mesh Quality and Mesh Smoothness

Several criteria can be used to measure the quality of a mesh such as minimum angle, maximum angle, minimum edge, maximum edge, minimum Jacobian, shape factor, etc. For this work, the minimum angle and shape factor are used as mesh quality criteria. The minimum angle criterion means that elements with small angles are considered to be of a worse quality than ones with larger angles. The shape factor criterion measures the likeness of an element to a reference equilateral triangle, and is given as,

$$SQ_i = \frac{4\sqrt{3}A_i}{\sum_{i=1}^{3} {l_i}^2} \tag{5.13}$$

where A_i is the area of the triangle, and $l_i (i = 1, 2, 3)$ are the lengths of the triangle's edges. However, in order to devise a mesh mapping appropriate for arbitrary boundary shapes in the physical domain, in addition to positive Jacobian for all cells, other measurable criteria must also be considered. In contrast to the traditional definition of mesh quality, which usually considers individual criteria of each element, smoothness of a mesh has a global definition. Thus, these two distinct measures of mesh quality and mesh smoothness may be contradictory for some cases. Indeed, a smoother mesh does not necessarily imply better mesh quality as we are going to show in this study. The smoothness criterion was introduced by (Arabi-Narehei, 2012) and will be used to compare meshes resulting from the two smoothing techniques investigated in this study. The mesh smoothness is quantified for each cell as,

$$SR_i = \frac{A_i}{max(A_n)} \tag{5.14}$$

where SR_i represents the smoothness ratio, A_i is the area of cell i and the denominator represents the maximum area of its adjacent cells. An ideal values for SR_i is as close as possible to one.

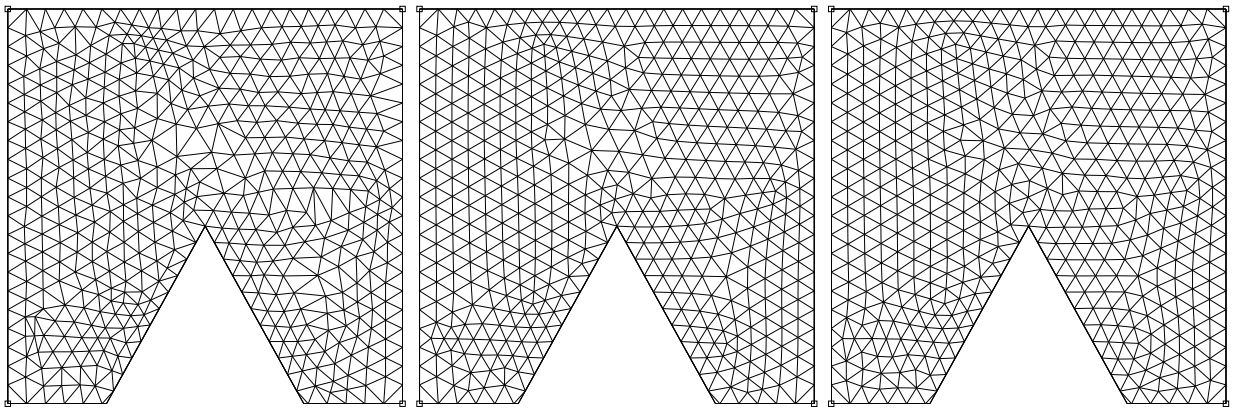
The global smoothness of a mesh is given by the Smoothness Factor (SF) of a mesh, defined in (Arabi-Narehei, 2012)

$$SF = \frac{1}{N_e} \sum_{1}^{N_e} min(SR_i, \frac{1}{SR_i}) \tag{5.15}$$

where N_e is the total number of elements in the mesh. The range of values for this factor is $0 < SF \leq 1$, and hence, the greater SF, the smoother the mesh.

5.5 Results and Discussion

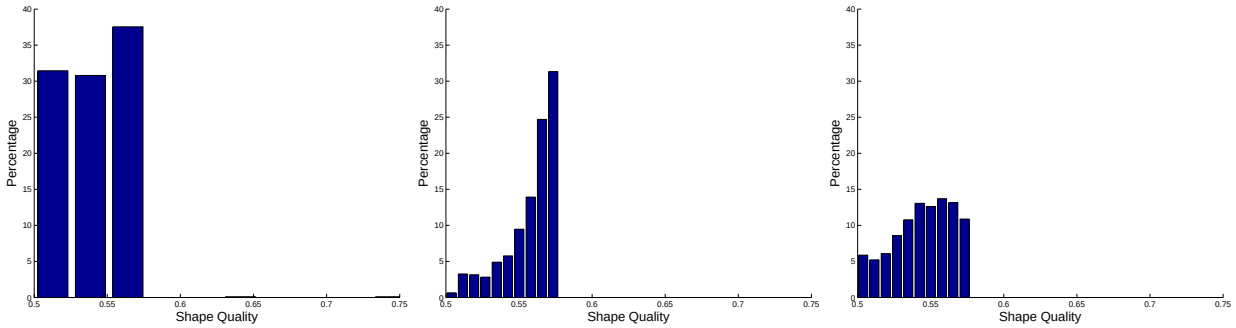
Two smoothing techniques (Winslow and barycentric) should were implemented and investigated in the context of their effectiveness. Numerous test cases for different geometries and grid sizes were performed. Two representative examples, a sharp spike and a slit inside a circle, will be used to illustrate the results. The effect of both smoothing techniques on an initial raw mesh, generated using a frontal unstructured grid procedure, is shown in Fig. 5.3. These qualitative results are quantified in Figs. 5.4 to 5.5 for the shape measure, minimal angle and smoothness ratio, respectively.



(a) Raw Mesh (b) Barycentric Method (c) Winslow Operator

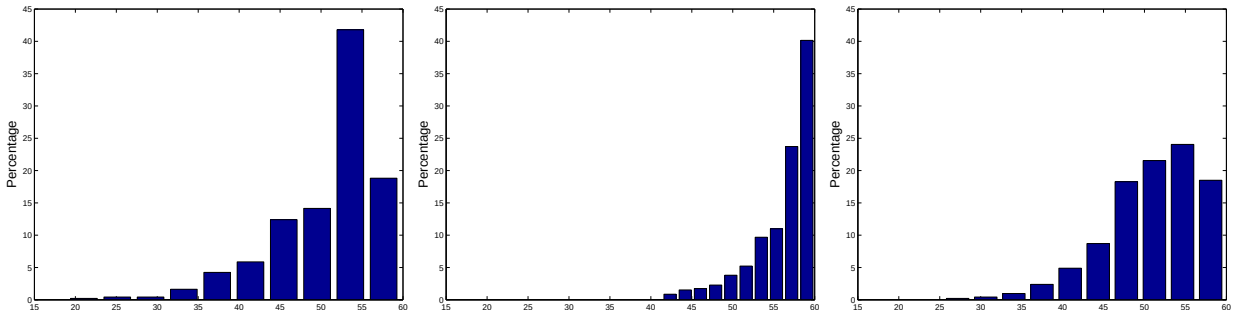
Figure 5.3 Comparison of two smoothing methods with the raw mesh

As can be seen from Fig. 5.4, the barycentric method gives a more satisfactory distribution from the shape factor viewpoint. This is also the case for the minimum angle criterion (Fig. 5.5), as the smallest angle for the barycentric method is around 40^o, while for the Winslow method this is around 25^o . However, from the smoothness point of view, it is clear that the Winslow operator gives better results than the barycentric method (Fig. 5.6).



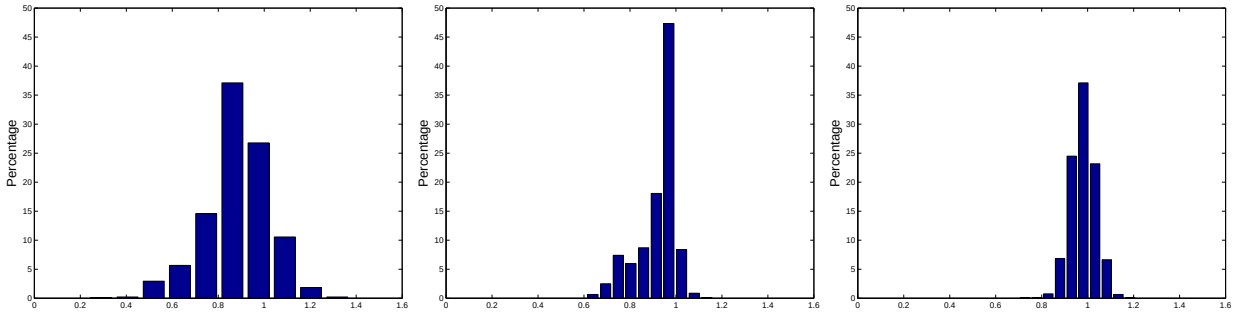
(a) Raw Mesh (b) Barycentric (c) Winslow

Figure 5.4 Comparison of mesh quality using the shape factor criterion



(a) Raw Mesh (b) Barycentric (c) Winslow

Figure 5.5 Comparison of mesh quality using the minimum angle criterion

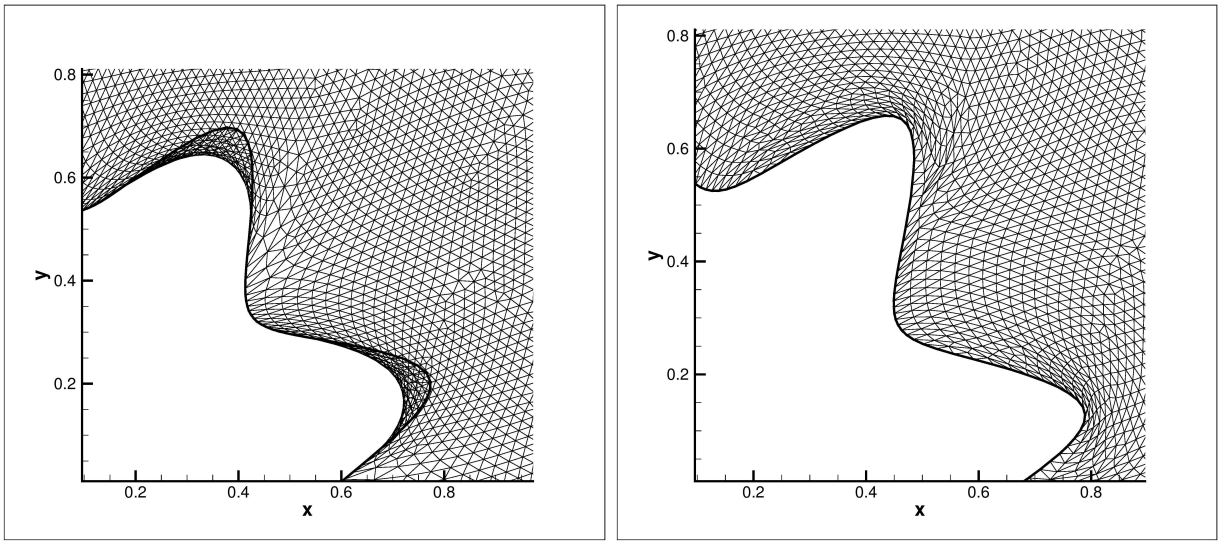


(a) Raw Mesh (b) Barycentric (c) Winslow

Figure 5.6 Comparison of mesh quality using mesh smoothness (SR)

Furthermore, for a configuration with strong curvature(s) the barycentric method fails to give a valid grid (Fig. 5.7). This is the greatest disadvantage of the barycentric method

versus Winslow‘s operator at a grid smoothing procedure.



(a) Barycentric Method (b) Winslow's Equation

Figure 5.7 Comparison of barycentric and Winslow's method to smooth a configuration with strong curvature(s)

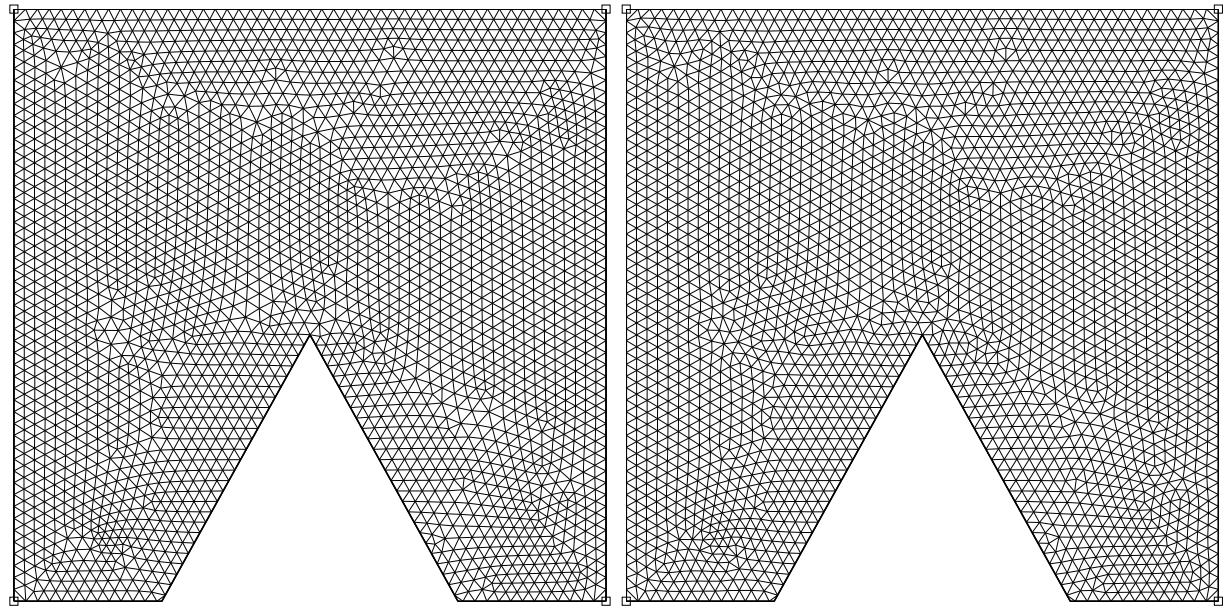
The comparisons were repeated for a finer grid to show the independence of the results from grid size (Fig. 5.8). As it can be seen again, the minimum angle is further improved using the barycentric method and the density of the elements with a minimum angle close to 60° is much higher than those with Winslow's operator. However, a better smoothness factor is again obtained with Winslow's operator.

Table 5.1 presents a quantitative comparison of smoothness factors for both grid smoothing methods and all presented test cases. It is observed that the smoothness factor for Winslow's method is higher than that resulting from the barycentric method. Similarly, for the spike geometry, with both fine and coarse grids, a better smoothness factor is also obtained using Winslow's operator. Similar conclusions apply for the circle with a "W" slit inside.

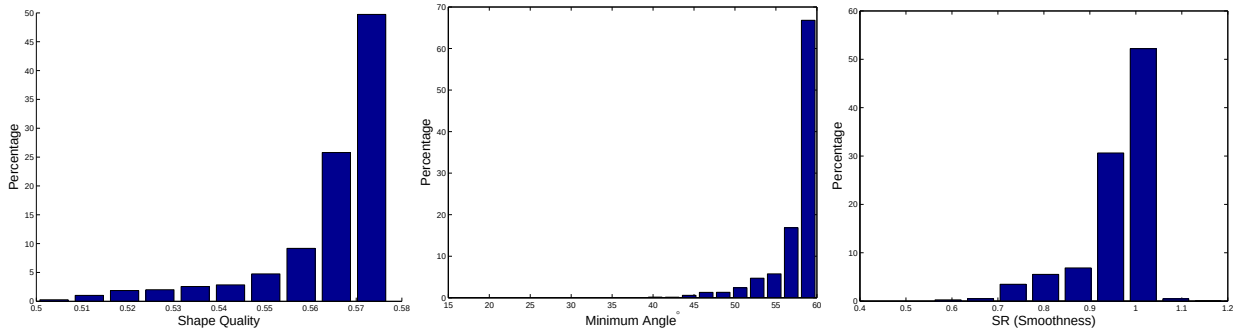
Table 5.1 Comparison of global Smoothness Factor for different mapping operators

Operator	Spike (coarse mesh)	Spike (fine mesh)	Slit
Raw Mesh	0.860 (Fig. 5.3(a))	0.882	0.865 (Fig. 5.9(a))
Barycentric	0.921 (Fig. 5.3(b))	0.951 (Fig. 5.8(a))	0.913 (Fig. 5.9(b))
Winslow	0.955 (Fig. 5.3(c))	0.971 (Fig. 5.8(b))	0.948 (Fig. 5.9(c))

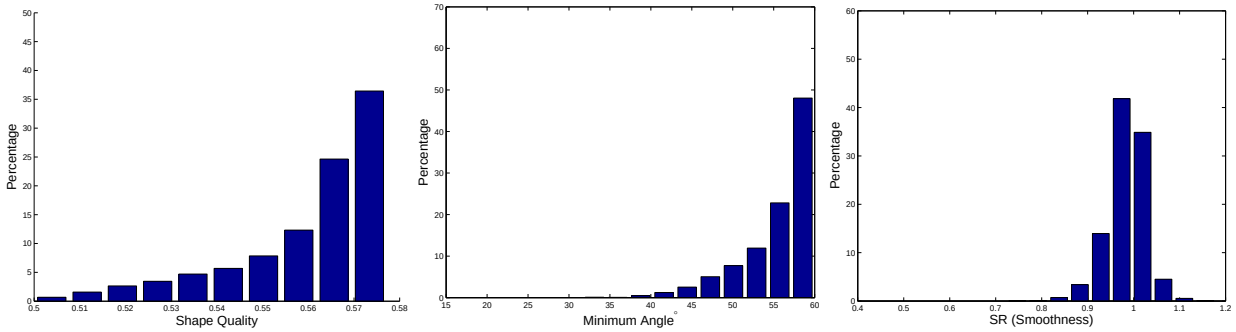
The comparisons were repeated for a finer grid to show the independence of the results from grid size (Fig. 5.8 and Tab. (5.1)).



(a) Barycentric (b) Winslow



(d) Barycentric



(g) Winslow

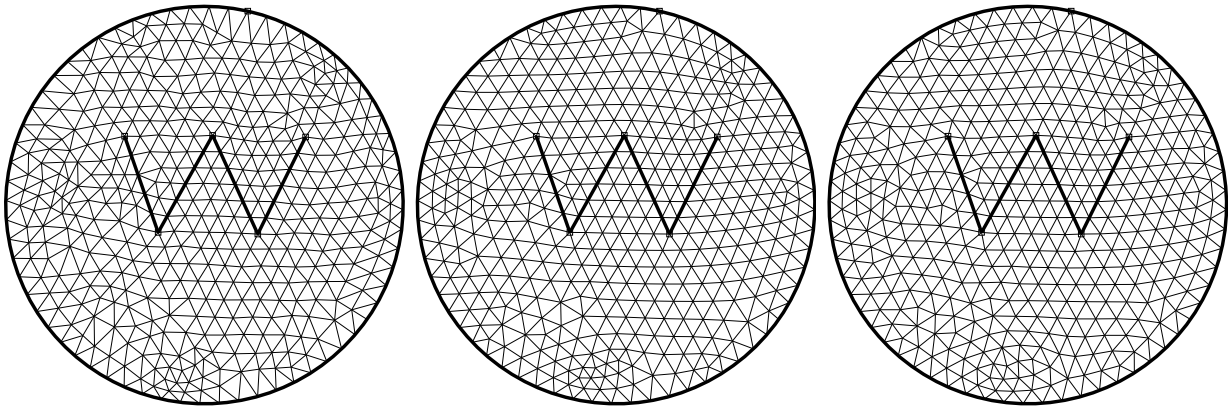
Figure 5.8 Comparison of mesh quality and minimum angle for a fine mesh

Another test case is a slit with a several sharp corners inside a circle (Fig. 5.9). Results show better mesh quality for the barycentric method but smoothness remains better for the Winslow method.

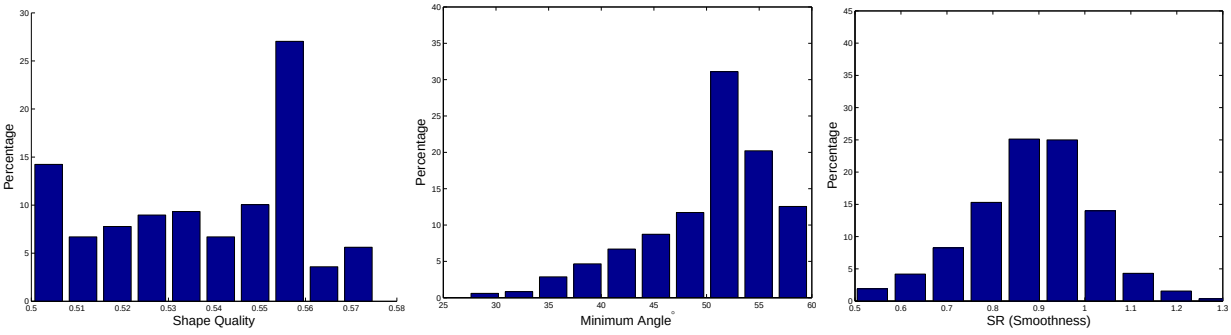
A final test was carried out to investigate the behaviour of each method applied to an invalid mesh shown in Fig. 5.10(a). Both techniques yield a valid mesh. However, the grid obtained by the Winslow operator is not identical to that obtained from an initial valid mesh, indicating dependence of the Winslow method on the initial mesh.

5.6 Application to Grid Motion

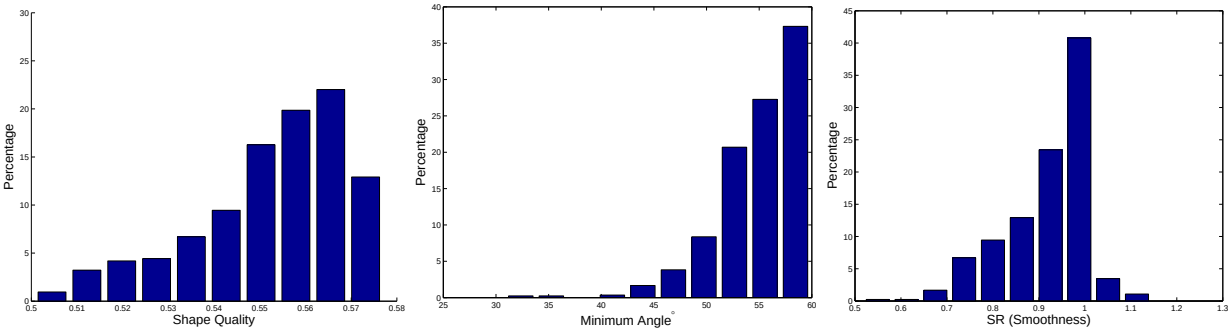
As discussed in chapter 3, one needs to devise a method in order to propagate the internal grid nodes into the field based on the new boundary condition imposed by the proposed grid motion approach in each time step. In (Illinca, 1996) this was done by specifying a mesh velocity to the moving boundaries which was then propagated to the internal grid points through the solution of a boundary value problem based on Laplace's equation. This insures that additional topological conditions are verified on the mesh movement, such as points initially on a boundary must remain on that boundary, while the points initially inside the domain cannot cross the border. In this work, this idea is implicitly employed by using Winslow's equation as it always guarantees the validity of the grid as well as the smoothness after implementation as it was studied in this chapter.



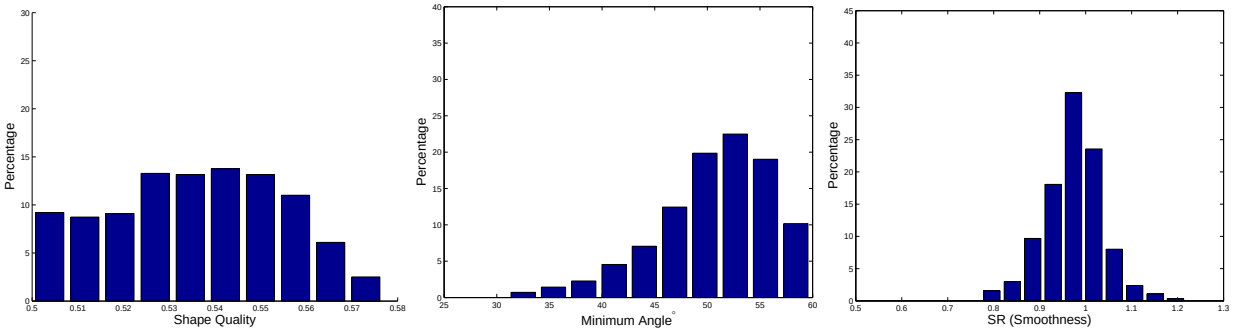
(a) Raw Mesh (b) Barycentric (c) Winslow



(e) Raw Mesh

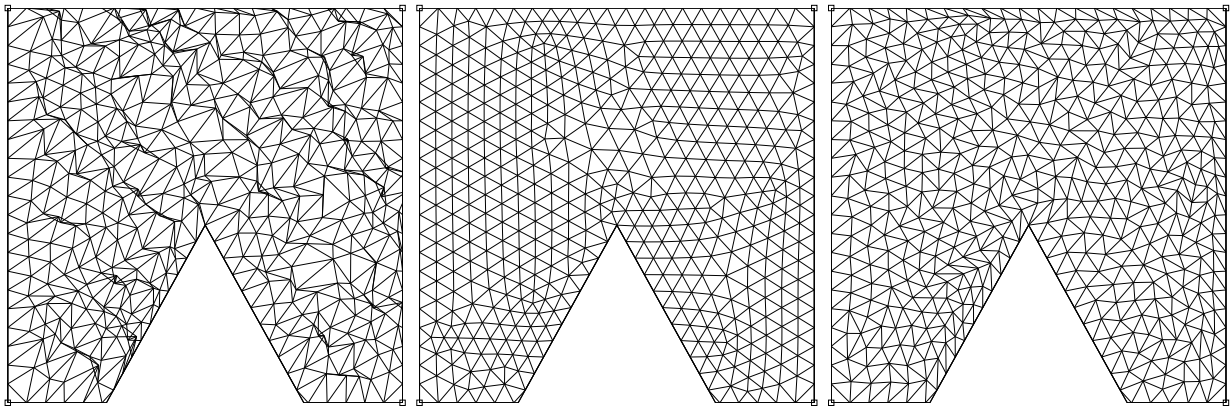


(h) Barycentric



(k) Winslow

Figure 5.9 Comparison of mesh quality and mesh smoothness for a slit inside a circle



(a) Invalid Mesh (b) Barycentric (c) Winslow

Figure 5.10 Investigation of the behavior of two smoothing methods for an invalid mesh

CHAPTER 6 APPLICATION TO A FLUID FLOW

Two principal components of a solution methodology for a flow field around a moving body are the grid velocity procedure and the flow solver. In order to study and evaluate the applicability of the proposed moving mesh methodology to real fluid flow problems, these two components will be combined in a flow solver based on a Lagrangian Eulerian formulation. In addition, the grid motion and evolution must obey the physical and geometrical conservation laws. The applied flow solver uses a generalized version of Roe's approximate Riemann solver.

The goal of this development is to validate the proposed moving grid methodology with a few selected moving boundary configurations.

6.1 Arbitrary Lagrangian Eulerian (ALE) methods

Generally, there are three viewpoints to simulate flow problems numerically : Eulerian, Lagrangian and Arbitrary Lagrangian Eulerian (ALE) approaches.

Eulerian Reference Frame : the grid is fixed in time and convective terms have to be computed. This is the typical framework used in the analysis of fluid mechanics problems. It allows the fluid to flow through the grid. However, it does not track the path of any individual particle.

Lagrangian Reference Frame : this is used most commonly in solid mechanics, free surface tracking and solving shock tube problems. It sets up a reference frame by fixing a grid to the material of interest, then, the grid follows the material as it deforms. In this method, the conservation of mass is automatically satisfied because the individual sections of the grid always contain the same (amount of) mass. It also defines the exact displacement of each particle, a feature that can be helpful in tracking motions in free surface and moving boundary problems. So, if a grid is mapped onto a fluid, as time evolves the fluid particles will travel independently of each other and diverge in space. This will cause the grid to distort excessively. Eventually, as the grid moves and deforms with the fluid velocity, it may overlap and become invalid. This deterioration of the geometric quality of the grid requires frequent remeshing.

Arbitrary Lagrangian Eulerian (ALE) : this is based on an arbitrary motion of the reference frame which combines the two previous approaches. It allows the grid to move with its own velocity independently of the fluid velocity. This results in a flexible, moving grid, whereby the fluid passes through the faces of the cells. This is helpful in

problems with large deformation of boundaries where the grid tracks the fluid and/or boundary (Aquelet et al., 2006; Belytschko et al., 1980; Donéa et al., 1977; Hassan et al., 2000; Hughes et al., 1981; Lesoinne and Farhat, 1996; Nkonga, 2000; Masud et al., 2007). When the grid deforms excessively and distorts the aspect ratio of the grid beyond an acceptable point, it adjusts the grid and measures the flux of the fluid or any other material during the adjustment of the grid. The difficulty in using the ALE approach is deciding how much to allow a grid to deform and how much flux to allow. This is usually done by setting a limit on the distortion of a segment of a grid and once it deforms past that limit, then that part of the grid is re-meshed.

The two general techniques that have been employed by various investigators are : (i) moving mesh proportional to the primary boundary motion, or (ii) solving the mesh motion through a proposed differential equation together with well-arranged boundary nodes as the boundary conditions.

(Illinca, 1996) has used that approach by imposing a velocity to the moving boundaries which was propagated to internal grid points. Additional topological conditions are imposed to the mesh movement, such as points initially on a boundary surface must remain on that surface, while the points initially inside the domain cannot cross the border.

6.2 Governing Equations

The integral form of the Euler equations for two-dimensional unsteady compressible flow may be written as

$$\frac{\partial}{\partial t}\int_{V(t)} U \,\mathrm{d}V + \oint_{S(t)} F \cdot \vec{n} dS = 0 \tag{6.1}$$

where U is the vector of dependent variables, F is flux tensor and $\vec{n}$ is the outward unit vector normal to the boundary $S(t)$, which encloses the time dependent volume $V(t)$. F_n, the component of F normal to the boundary $S(t)$) and U are defined as,

$$F_n = \begin{pmatrix} \rho(u_n - w_n) \\ \rho u(u_n - w_n) + P n_x \\ \rho v(u_n - w_n) + P n_y \\ \rho E(u_n - w_n) + P u_n \end{pmatrix}, \qquad U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} \tag{6.2}$$

where ρ is the density, u_n and w_n are, respectively, the components of the fluid velocity and that of the moving mesh, normal to the surface on which F_n is calculated. E is the specific

energy and P is the pressure.

Equations 6.1 and 6.2 represent the conservation of mass, momentum and energy. In addition, an equation of state is required. For an ideal gas this is

$$p = (\gamma - 1)\rho e \tag{6.3}$$

where γ is the ratio of specific heats and $e = E - \frac{1}{2}u.u$ is the specific internal energy.

Eqs. 6.2 differ from their counterpart for stationary grids by the term w_n which accounts for the grid velocity. This velocity component, w_n, should be calculated based on Geometric Conservation Laws (GCL) in order to prevent any error arising from mesh movement. This is an often overlooked issue in moving grids. The discretization of the GCL was proposed for the first time by (Demirdžić and Perić, 1988) for a finite-volume scheme. The major point is to convect the conservative variables by $(\vec{u} - \vec{w})$ instead of $\vec{u}$, so that $\vec{u}$ is the material velocity vector and $\vec{w}$ is the arbitrarily specified facial velocity.

These consist in two equations which state that cell volumes must be bounded by their surfaces (Surface Conservation Law, SCL) and that a volumetric increment of a moving cell must be equal to the sum of changes along the surfaces that enclose the volume (Volume Conservation Law, VCL). An incorrect rate of the convective velocity or additional source may be encountered when GCL are not satisfied. The volumetric conservation law is given by

$$\frac{\partial V}{\partial t} - \oint_B w.dS = 0 \tag{6.4}$$

while the surface conservation law is

$$\oint_B a.dS = 0 \quad if \quad \oint_B dS = 0 \tag{6.5}$$

where the control volume V is bounded by the face B. The discrete form of Eq. 6.4 from time t_n to $t_n + \triangle t$ gives

$$V^{n+1} - V^n = \sum_i \int_{t_n}^{t_n+\triangle t} \int_{B_i} w.dSdt = \sum_i \triangle V \tag{6.6}$$

where the subscript n denotes the value of the time t_n, and the value of $\triangle V_i$ indicates the volumetric increase (or decrease) along the face B_i. This equations states that, during a time interval, the increased volume in a control cell equals the summation of the volumetric increase along its face which is the definition of GCL. Associated with the facial volume

increment $\triangle V$, the relevant facial velocity w_n during a time step $\triangle t$ is defined as,

$$w_n = \frac{\Delta V}{S\Delta t} \tag{6.7}$$

where S represents the length of the side which can be evaluated either at t_n or t_{n+1} or a linear combination of these two, as the resulting velocity is always consistent (Reggio et al., 1992). The calculation of this is shown in Fig 6.1. For sub-control volume $oa1bo$ in time t is changed to $o'a'1b'o'$. Based on this, the facial velocities normal to the surfaces oa and ob are

$$(w_n)_{oa} = \frac{\Delta V_{oa}}{S_{oa}.\Delta t} \qquad (w_n)_{ob} = \frac{\Delta V_{ob}}{S_{ob}.\Delta t} \tag{6.8}$$

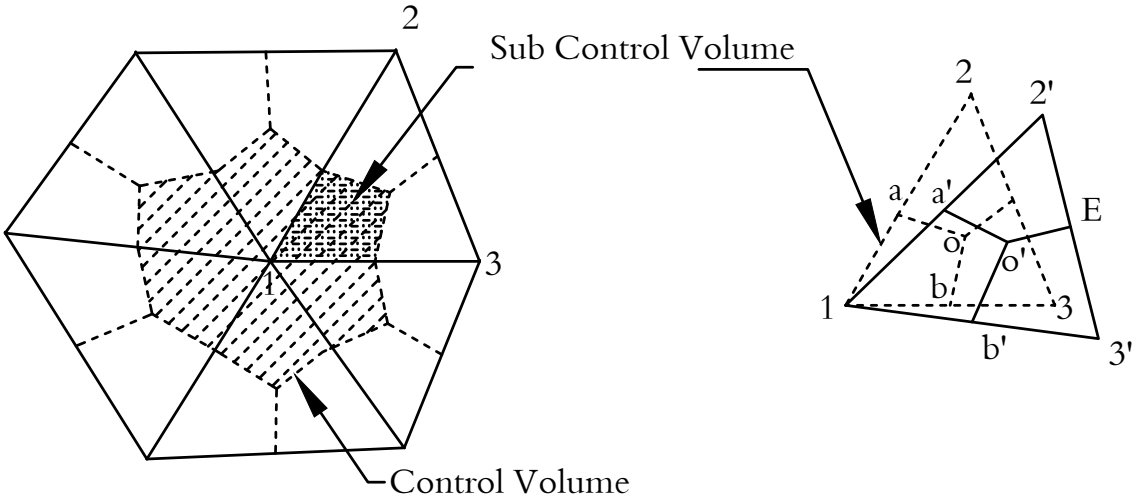


Figure 6.1 A control volume deformation during a one time step mesh motion

6.3 The Roe Scheme

In the present work, Roe's scheme has been used to calculate fluxes for the Euler set of equations. A comprehensive explanation about this method can be found in (Reggio et al., 1992). For the sake of completeness, the extension of this scheme for moving grids is briefly presented here. For a fixed grid, following (Roe, 1981; Reggio et al., 1992), the flux between adjacent cells can be written as,

$$F^f(Q) = \frac{1}{2}\left[F_R^f(Q) + F_L^f(Q) - \sum_{i=1}^{4} \alpha_i \left|\Lambda_i^f\right| e_i\right] \tag{6.9}$$

where the subscript f has been used to denote fixed mesh and the subscripts R and L have been used to characterize the right and left states at a given interface. F is the total flux across an interface during a time step $\triangle t$. Q is defined as,

$$Q = u_n S \triangle t \tag{6.10}$$

The expressions for F and the flux eigenvalues in this equation are well documented in (Roe, 1981). Once the flux variables are known, the properties U are advanced to the new time position $n+1$ using a finite volume approach given by

$$(U^{n+1} - U^n)V^n = -\sum_{k=1}^{Nsides} F^f(Q_k) \tag{6.11}$$

In the moving grid case, the grid motion only affects the convective variables. In particular, the Q term in Eq. 6.11 becomes,

$$Q^m = (u_n - w_n)S \triangle t \tag{6.12}$$

where the superscript m denotes a moving mesh. Using Eq. 6.7, this can be rewritten as,

$$Q^m = u_n S \triangle t - \triangle V = Q^f - \triangle V \tag{6.13}$$

With this definition, the flux for a moving mesh is

$$F^m(Q) = \frac{1}{2}\left[F_R^m(Q) + F_L^m(Q) - \sum_{i=1}^{4} \alpha_i \left|\Lambda_i^m\right| e_i\right] \tag{6.14}$$

and the flow properties can be calculated in an explicit manner, as

$$U^{n+1} = \frac{V^n}{V^{n+1}}\left[U^n - \frac{1}{V^n}\left(\sum_{k=1}^{Nsides} F_k(Q_k^m)\right)\right] \tag{6.15}$$

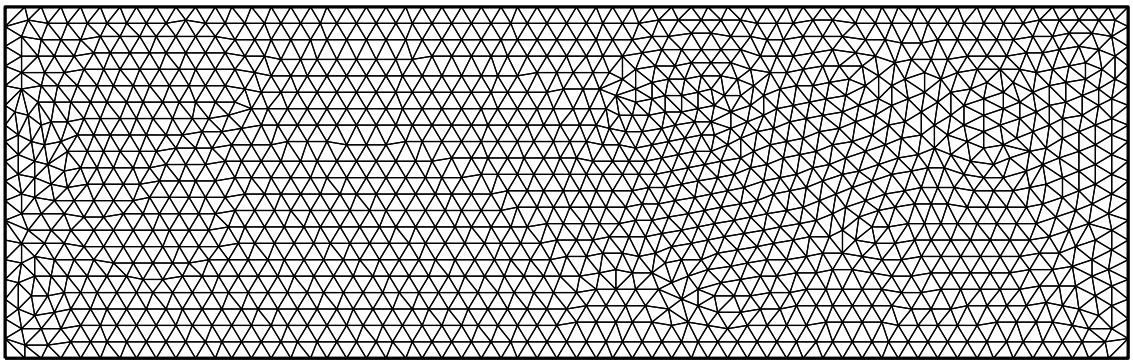
6.4 Solver Validation of Sliding Mesh Methodology

In the present work, an unsteady Euler solver code developed for stationary grids by (Masatsuka, 2013) has been modified to be used for dynamic grids [1]. The scheme is node-centered finite volume based on the finite element method (FVBFD) for unstructured grids to solve the Euler equations. Gradients are calculated by unweighted least-squares method,

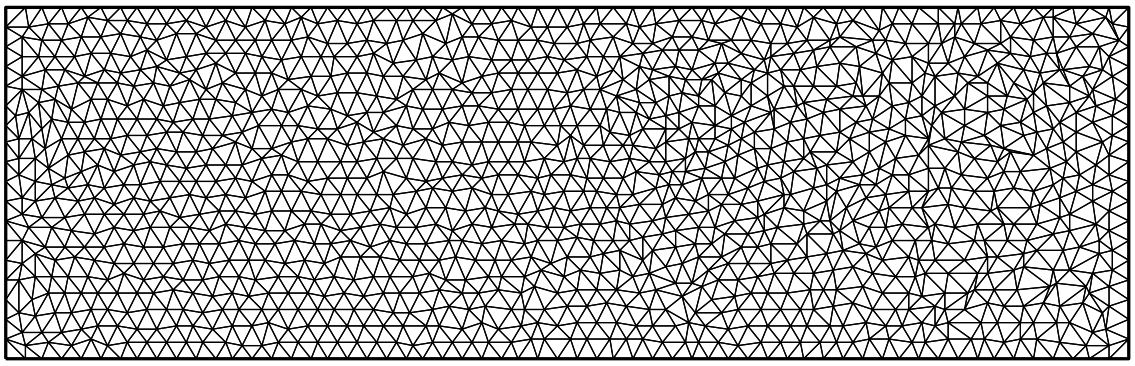
1. The code for fixed grids and validations can be found at `http://www.cfdbooks.com/`

and a 2-stage Runge-Kutta global time-stepping towards the final time is applied. The code was extended by adding the relevant facial velocity, w_n in Eq. 6.2, to handle dynamic grid cases as discussed in the previous sections.

To validate the modified flow solver for moving grids, a common test introduced in (Reggio et al., 1992) is performed by simulating the flow field in a rectangular cavity with a moving unstructured grid shown in Fig. 6.2(a). The fluid is at rest and the grid is set in motion randomly while keeping a fixed connecivity, and in a manner as to always give a valid grid. Fig. 6.2(b) shows the grid pattern after 20 steps.



(a) Initial grid



(b) Grid pattern after random motion

Figure 6.2 Verification of the GCL implementation by random motion of the grid

Fig. 6.3 illustrates the computed isovalues of the ρ field using single precision, which validates the correct implementation of GCL and shows that this random grid motion does not significantly affect the stationary flow.

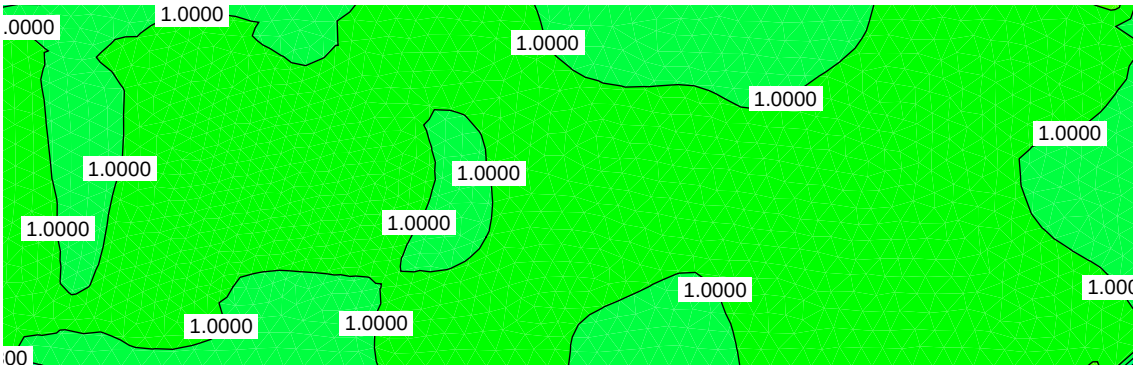


Figure 6.3 Stationary Flow field resulting after 20 steps of random motion of the grid

6.5 Applications

The purpose of this section is to demonstrate the capability of the proposed grid motion procedure in representative fluid flow problems. This includes the basic idea of a moving mesh with fixed connectivity and its associated data structure, using the proposed mesh sliding procedure described in Chapters 3 and 4. The grid management procedures are carried out in physical space with the body moving along a defined trajectory within a reference grid, and the proper grid velocity computation for internal cells. These elements are integrated into a global solution procedure using a finite volume ALE flow solver based on Roe's scheme modified to verify the GCL conditions.

It is necessary to mention that all the simulations are carried out to test the validity and compatibility of the proposed moving grid methodology with the flow solver and to verify that the coupling is correctly implemented. No efforts were made to generated accurate solutions as these would require much finer grids than those used for the current study.

Translation of an Airfoil This section presents the results of an application to simulate the unsteady flow past a moving airfoil. A typical airfoil geometry included with the path of the motion (trajectory) is shown in Fig. 6.4. The leading and trailing nodes of the airfoil slide along the trajectory, insuring that the motion is along the trajectory.

A slip boundary condition is set on the upper and lower parts of the airfoil, and free stream conditions are applied at the entrance (left) of the domain. Outlet boundary conditions are applied at the exit (right) of the domain where the gradient of all variables are set to zero except the pressure. These are summarized in Table 6.1.

The domain is discretized with approximately $18,000$ triangular elements.

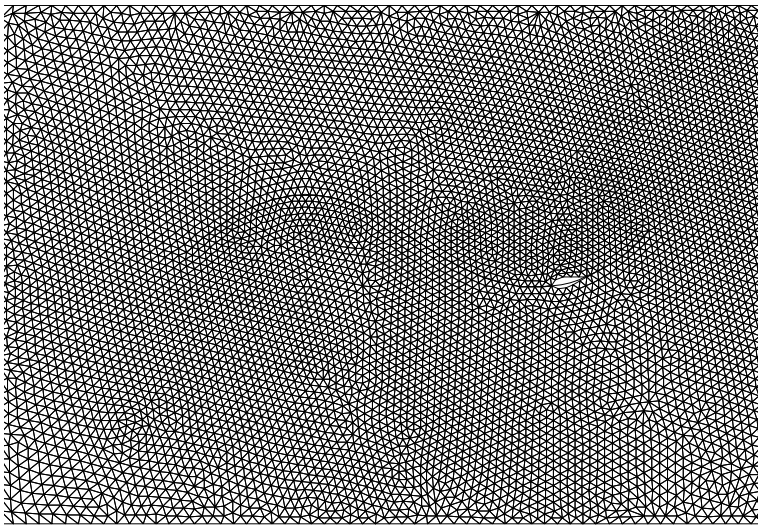


Figure 6.4 Airfoil moving along a trajectory for conditions given in Table 6.1

Table 6.1 Given values at the entrance of the domain for a free stream boundary condition

M_{inf}	γ	ρ_{inf}	P_{inf}	α (angle of attack)
0.3	1.4	1 (for air)	$1/\gamma$	0°

A dimensionless speed (v_b) equal to 0.1 is set for the moving body. The motion starts from point A and ends at the point B. The length of displacement (s) is the curvilinear distance along the trajectory between these two nodes. Since the total displacement and the average velocity of the body (v_b) is known, the duration of the entire motion (t) can be determined from,

$$t = \frac{s}{v_b} \tag{6.16}$$

The time step, Δt, is calculated by the solver for each step and this is used to determine the displacement (Δs) for the next time step,

$$\Delta s = v_b \Delta t \tag{6.17}$$

This shows the importance of managing the zipping method independently of the flow solution as the displacement may vary for each step. Fig.6.5 shows the body-grid configuration and velocity field at three different positions during the motion. The connectivities are fixed during the entire motion, as it can be seen, by tracking node #1001 [2].

2. The full motion can be watched on `https://youtu.be/WEd6KwLZwjU`

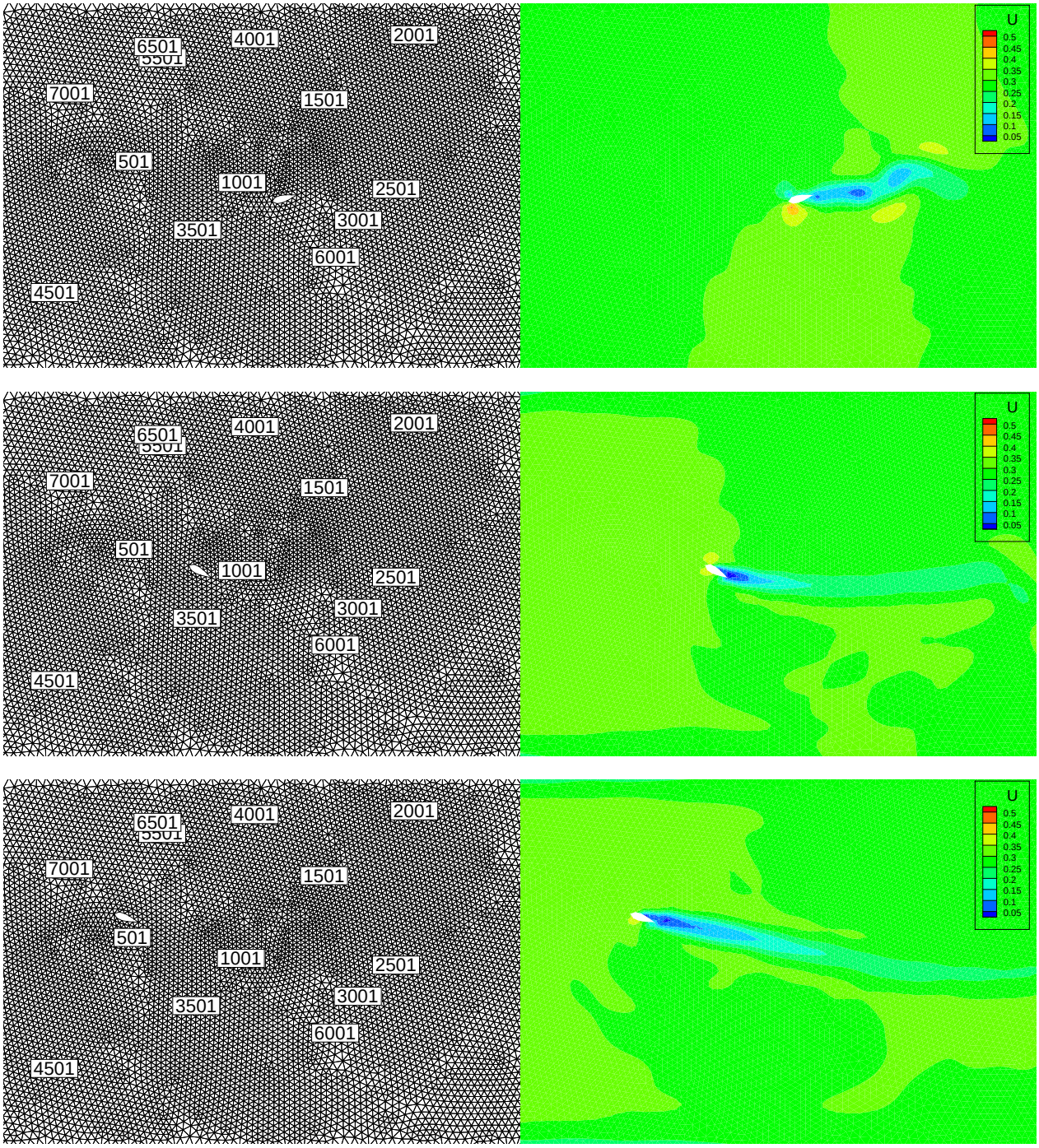


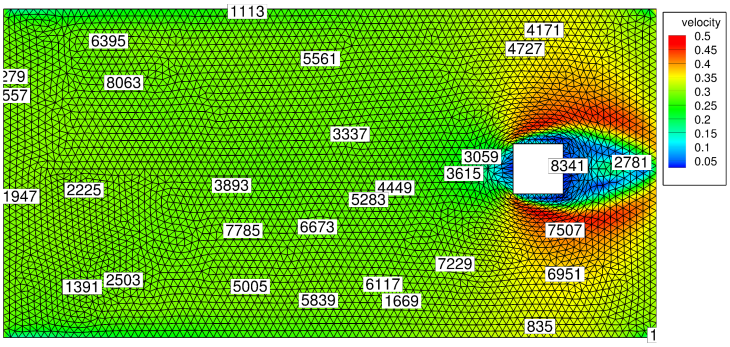
Figure 6.5 Grid and velocity field for an airfoil moving along a trajectory for three time steps, using the conditions in Table 6.1

Moving Square To verify the proposed method for a composite or hybrid grid, the Euler equations are solved to simulate the flow around a moving square. In this configuration, the body (obviously with sharp curvature) moves through a mesh with fixed connectivity using the hybrid grid sliding extension presented in Chapter 4. As discussed, because of the large variations of the curvature of the moving boundaries, the sliding method cannot be used solely with a single block grid and requires a composite grid made up of a grid attached to the body, making the transition to the outer reference mesh. This is illustrated in Figs. 6.6 and 6.7 and the evolving grid and velocity field are shown for eight time steps. The mesh is shown in background in order to show the fixed connectivity and the multiple/hybrid grid configuration of the square-fitted mesh sliding inside the reference mesh. The boundary conditions used in the previous example are applied for this case, that is a free stream inlet boundary condition is equal to 0.3 and the density is equal to one. The domain is discretized with around $15,000$ triangular elements and the moving block contains around 1000 triangular elements [3]. The moving square is immersed in a circular domain and it is this circular domain which slides through the reference mesh. This can be seen in Figs. 6.6 and 6.7 by tracking node #8341 which moves with the square since it is inside the circular domain, and at the same time node #3337 which is moves inside the reference grid

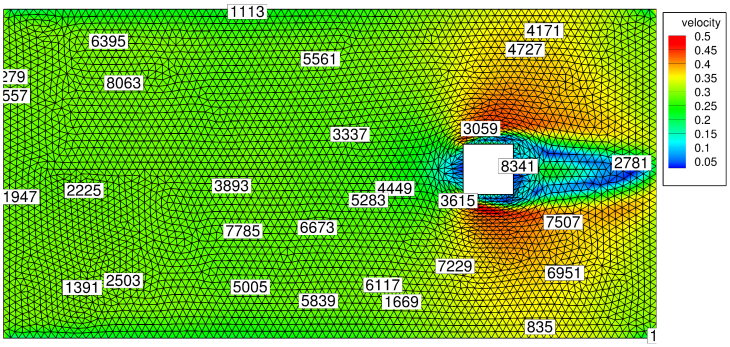
Oscillating Flap The next test case presents an oscillating flap at the tail of an airfoil which can be considered a representative motion for this type of applications. Slip boundary condition is set for both the airfoil and the flap and free stream boundary condition is applied at the entrance (left side) of the domain (the values are given in table 6.1. Outlet boundary conditions is applied at the exit of the domain where the gradient of all variables (except pressure) are set equal to zero. The flap is oscillating with a interval of $\alpha = 22.5°$ starting from $\alpha = 0°$. The grid is rotated with an angular step of $\alpha = 0.5°$.

This example is used to illustrate the ability to manage the sliding method for rotational motions with multiblock application. As it can bee seen from Fig. 6.8(a), the grid is much finer around the oscillating flap than the far field which is the objective of introducing the moving composite/hybrid grid extension of this work . This is required in most engineering flow simulation cases where one needs to control concentration in a flow field. Fig. 6.8(b) shows the initial solution at time $t = 0$. Figure 6.9 shows the velocity field and the dynamic grid at three time steps. It should be noticed that the nodes on the main grid are always fixed (e.g. node #337) while for the nodes inside the oscillating block (containing oscillating flap) the positions are continuously changing (e.g. node #5797) [4].

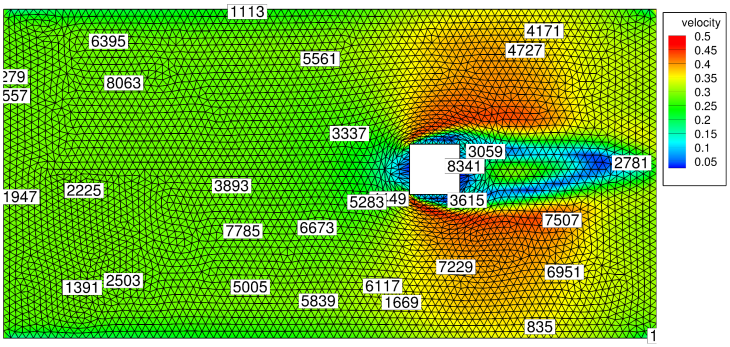
3. The full motion can be watched on `https://youtu.be/mKabX4EmJzM`
4. The full motion can be watched on `https://youtu.be/WJyj09Gyaww`



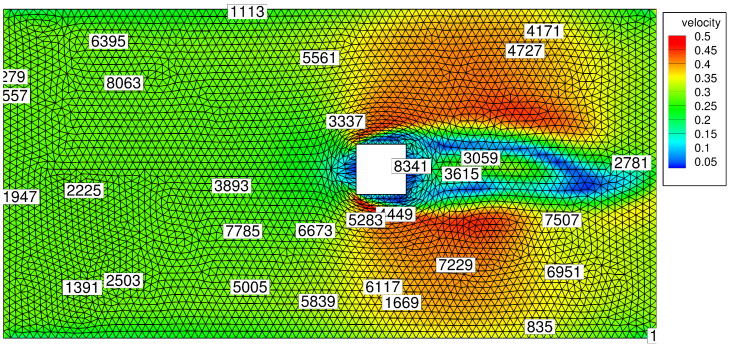
(a) $l = 0$



(b) $l = 1.5$

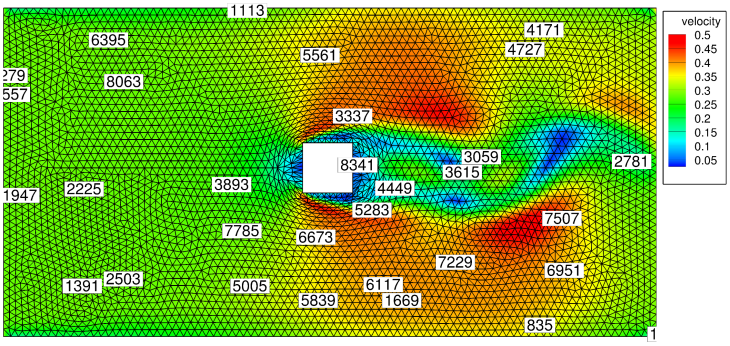


(c) $l = 3.0$

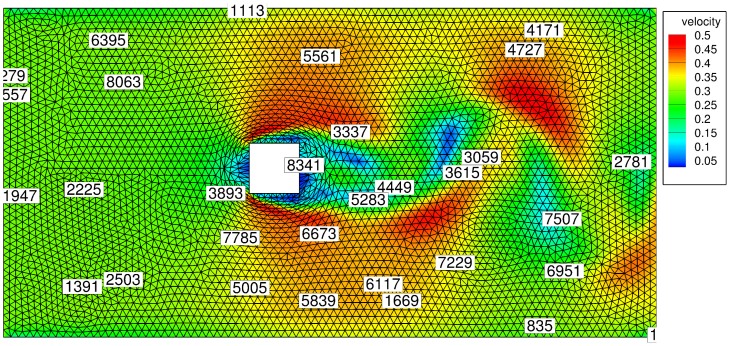


(d) $l = 4.5$

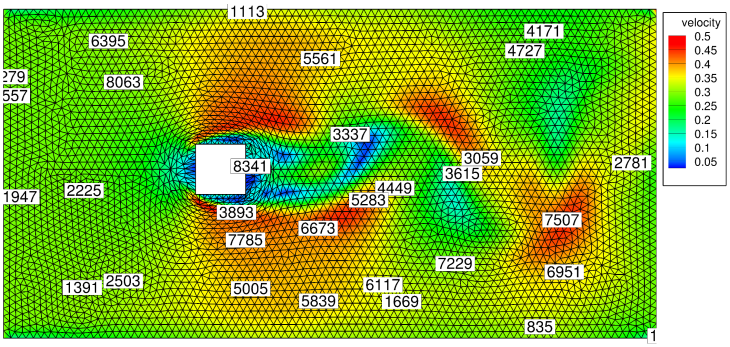
Figure 6.6 The velocity field of a moving square using zipping method for a multiblock test case



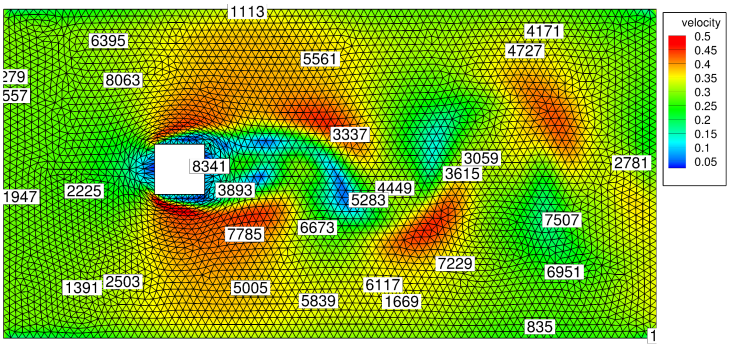
(a) $l = 6.0$



(b) $l = 7.5$

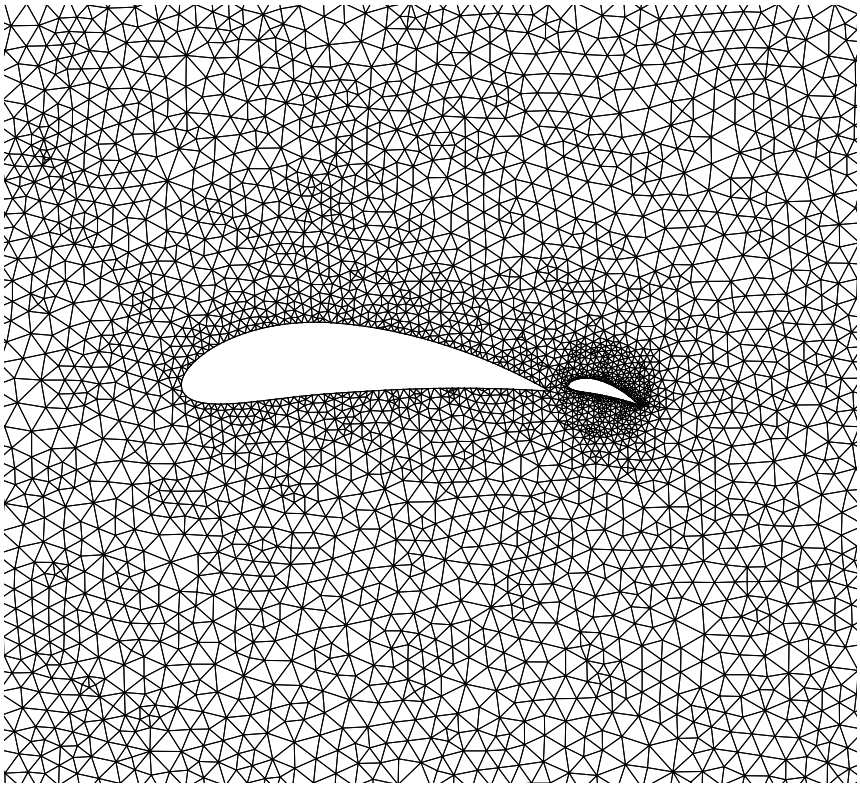


(c) $l = 9.0$

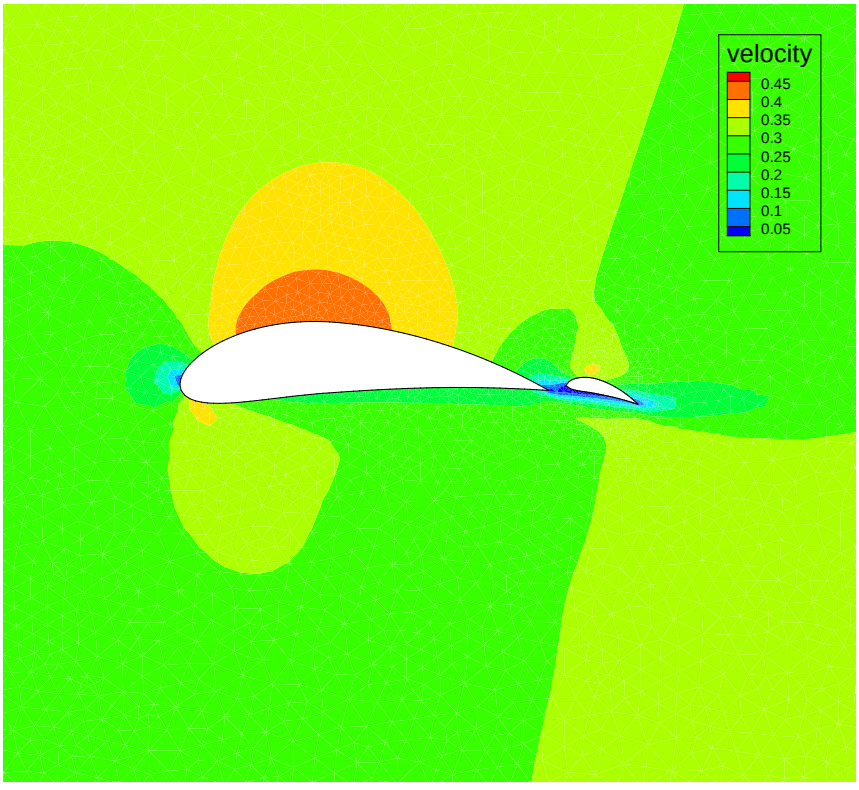


(d) $l = 10.5$

Figure 6.7 The velocity field of a moving square using zipping method for a multiblock test case



(a)



(b)

Figure 6.8 Grid and velocity field of an airfoil-moving flap configuration at $t = 0$

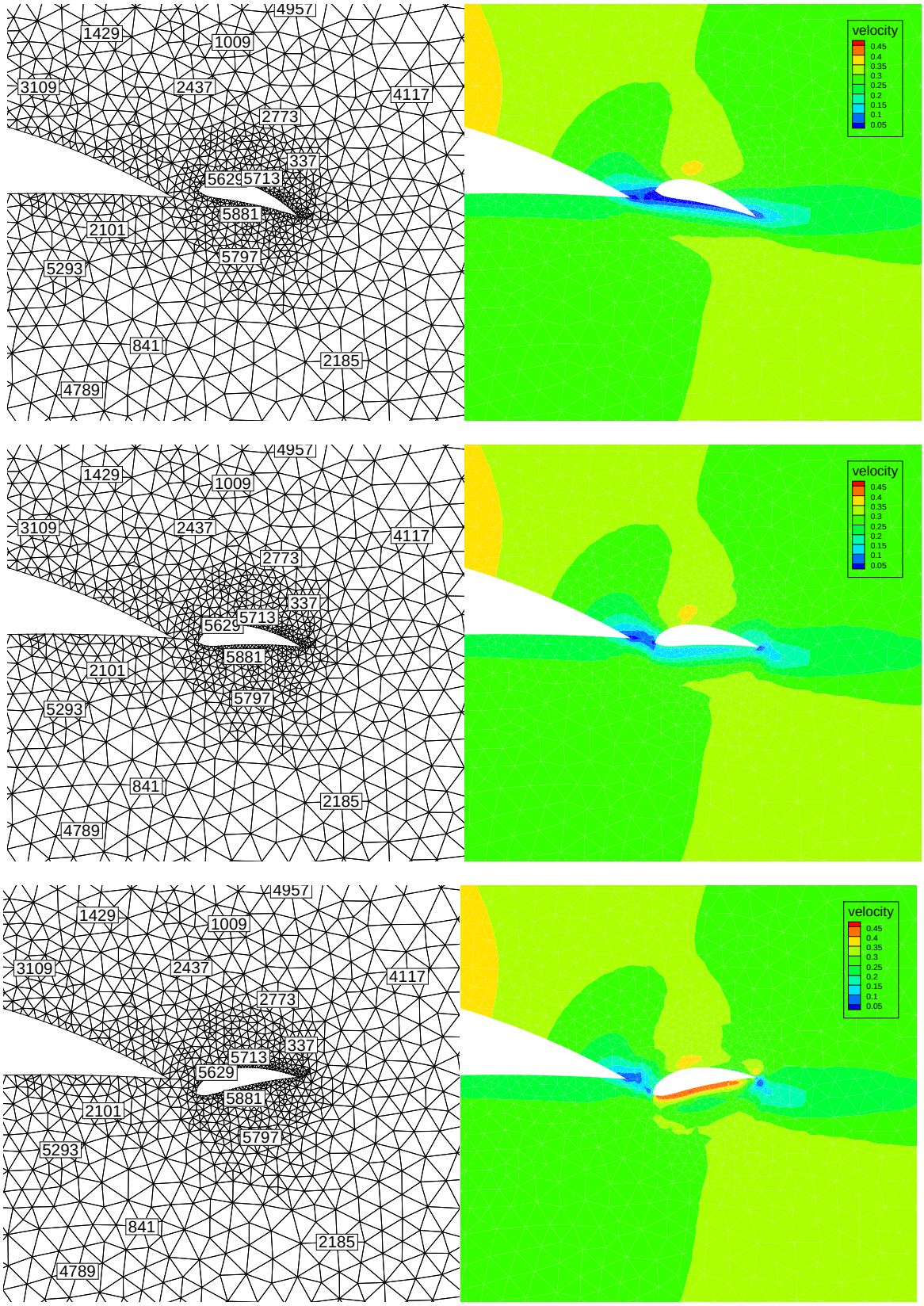
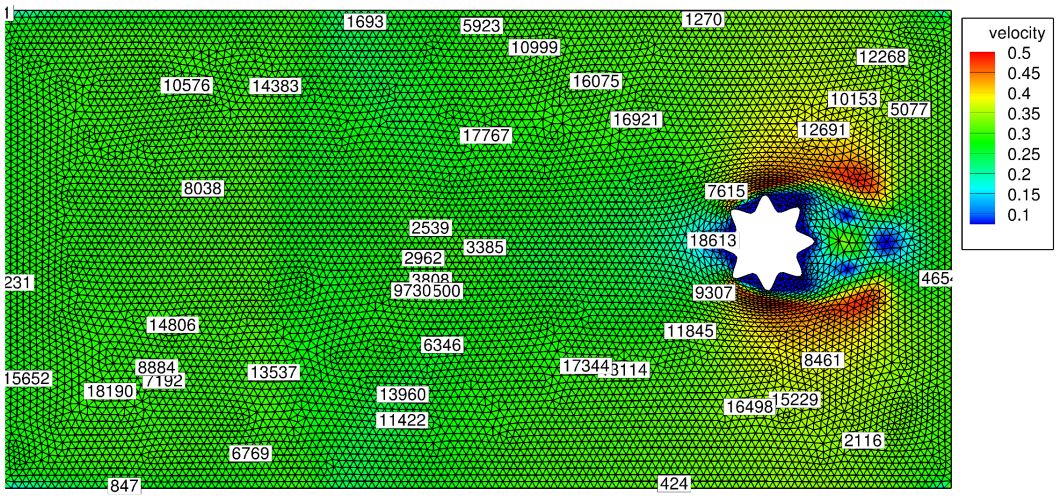


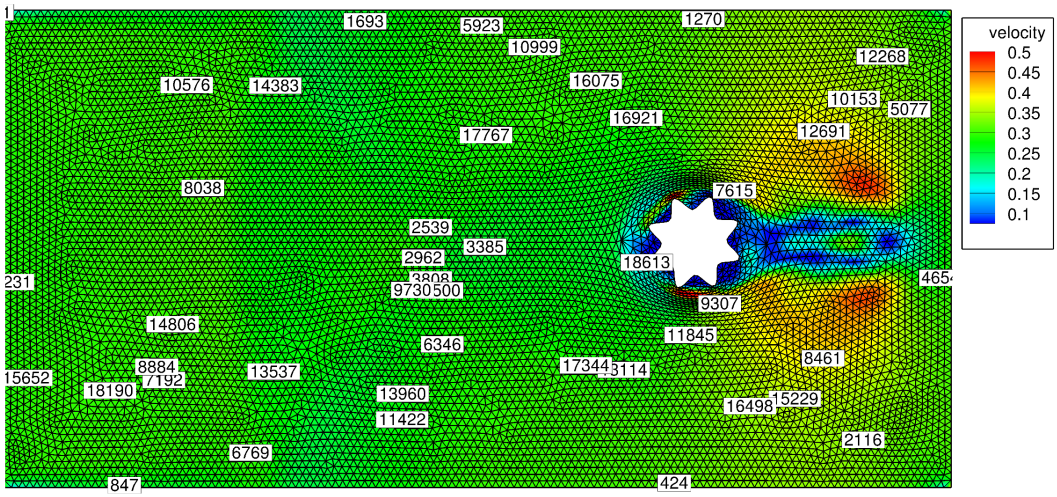
Figure 6.9 Grid and velocity field of an airfoil-moving flap configurationat three time steps

Complex Motion of Petals The last case presents the calculated fluid flow for a combination of translation and rotation of a complex geometry consisting of several petals, such as previously validated in Section 3.6.3. The same flows and boundary conditions and initial values as for the previous examples are used. The domain is discretized with around 35,000 triangular elements for the outer reference grid, and the moving inner block contains around 2500 cells. A total rotation of $\alpha = 180°$ is set for the moving petals superposed on the translation motion. This can be seen by tracking node #18613 in Figs. 6.10 and 6.11. This node rotates counter clockwise about the center of the body, as it belongs to the rotating inner mesh, while it translates and completes a full rotation [5]. The simpler translation motion of cells in the outer reference grid can similarly be observed by tracking appropriate nodes such as #16921. The motion of Node #7615 is particularly interesting as it lies almost at the limit between the outer and inner reference grids.

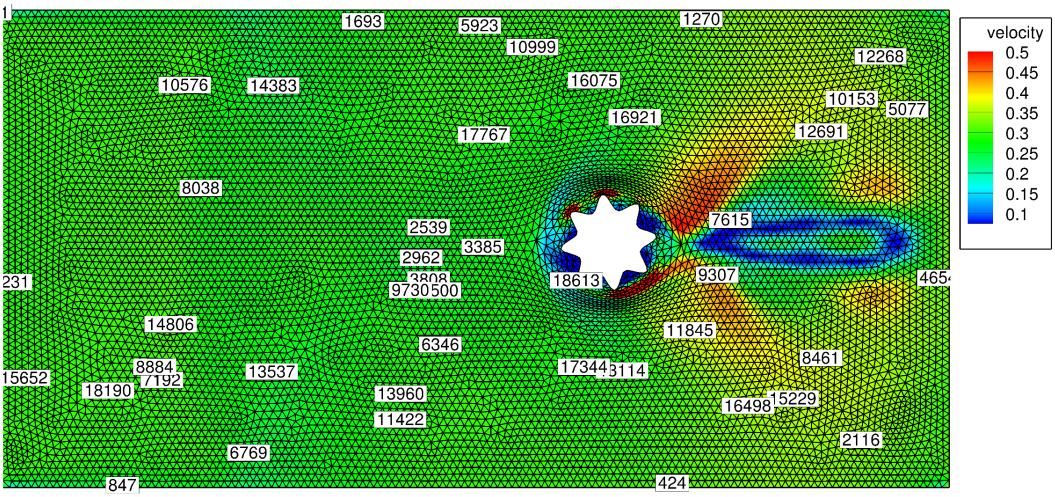
5. The full motion can be watched on `https://youtu.be/TkxDvH82dW0` and `https://youtu.be/2EyvPdD-U7Q`



(a) $l = 0 \quad \alpha = 0°$

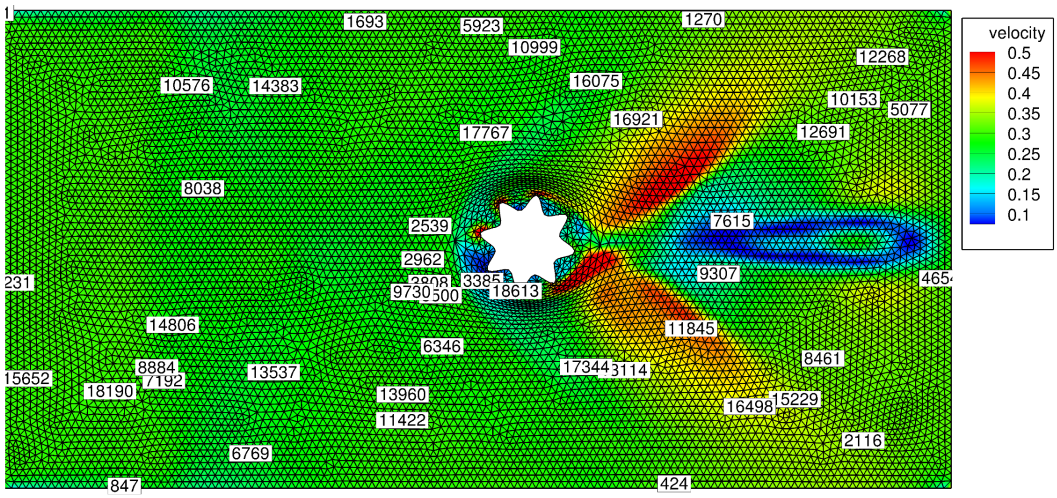


(b) $l = 1.5 \quad \alpha = 30°$

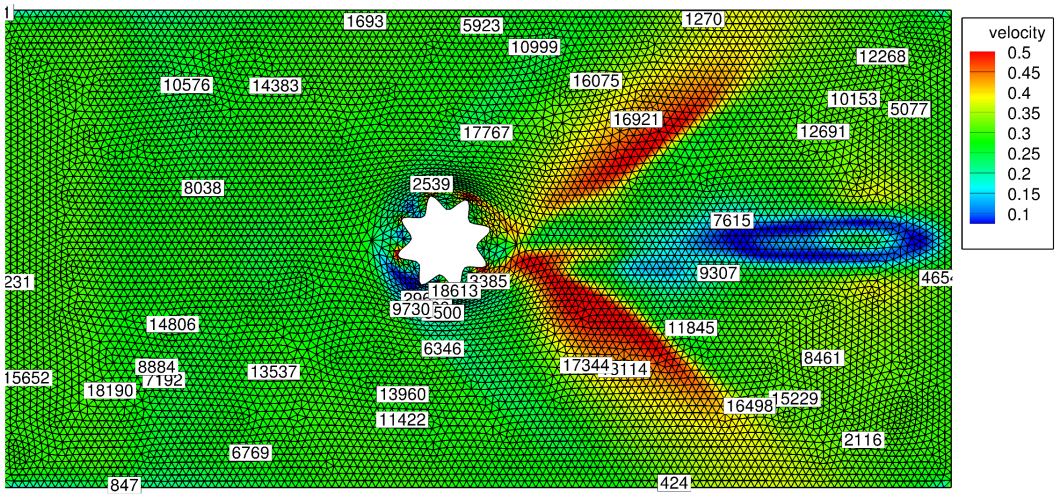


(c) $l = 3.0 \quad \alpha = 60°$

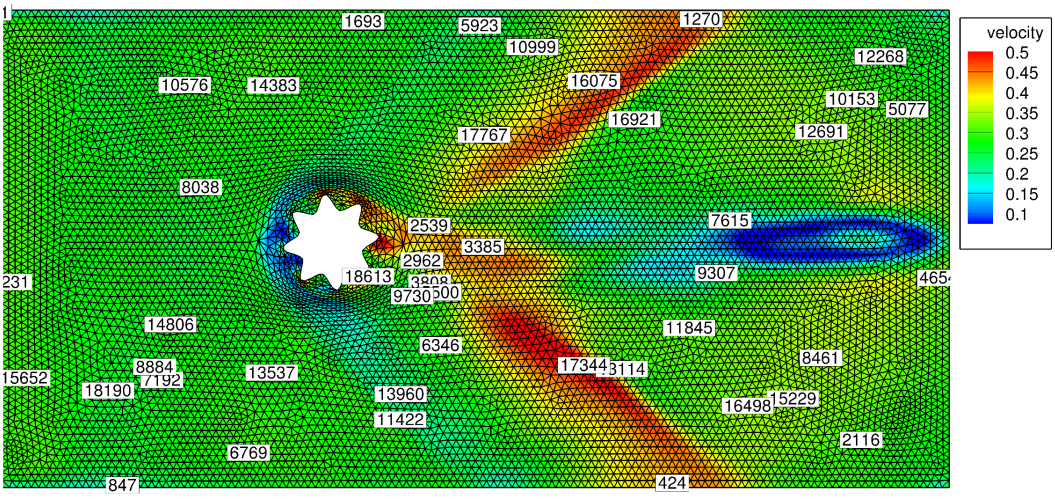
Figure 6.10 Grid and velocity field of a rotating-translating complex body, $0 < \alpha < \pi$



(a) $l = 4.5 \quad \alpha = 90°$



(b) $l = 6.0 \quad \alpha = 120°$



(c) $l = 8.0 \quad \alpha = 160°$

Figure 6.11 Grid and velocity field of a rotating-translating complex body, $\pi < \alpha < 2\pi$

CHAPTER 7 CONCLUSION

Generally, moving mesh generation is a conflicting situation between maintaining good mesh quality during time evolving boundaries and computing resources. The study of currently available approaches shows that this is still a challenging problem for real engineering applications from the point of view of

— boundary geometric complexity and large deformations of the domain ;

— algorithm complexity and robustness.

7.1 Contributions

The main contribution of this thesis is the development of a new body-fitted method for dynamic grid generation that addresses these issues as follows :

A new mesh at each time versus one mesh for all time : the moving mesh model is that of a body moving through a reference grid where the number of nodes is constant and the connectivity is fixed. The many advantages resulting from this approach can also be obtained by Overset Grids, Universal meshes or Immersed Boundary Methods (Chapter 2), in, however, varying degrees (Section 2.4). In the present work, these were achieved in a globally more efficient and robust manner. These resulted from an in depth study of the sliding method, where a number of critical extensions have been proposed and implemented avoiding several shortcomings (e.g. managing the moving procedure inside physical space instead of computational space in order to have better compatibility between the moving object and the reference grid and also introducing a trajectory for an automatic arrangement of the nodes along the movement). In addition, a new data structure was devised to make the proposed method ready and compatible with a general class of finite volume flow solvers based on an ALE formulation.

Grid management in physical space : it was found that managing the grid through procedures in physical space contributed to maintain the initial mesh quality throughout the boundary displacement. This is the case for schemes such as Overset Grids and Universal meshes. In the proposed method, this is further improved by the introduction of an explicit moving path which maintains the initial grid quality generated around this trajectory during the motion. It was found that the reference grid remains a compatible space filling mesh for the evolving configuration, and automatically insures valid nodal valence and edge arrangement around the moving object at leading and trailing edge

separation. In addition, the associated data structure significantly reduces the computation cost for grid management operations as only the nodes and elements in contact with the moving boundary are involved.

Implementation for translation and rotation : these procedures and associated data structures were combined into an overall grid moving algorithm and applied to achieve grid motion of several representative geometries in both translation and rotation. The grid displacement and subsequent mesh velocity were computed using an improved smoothing procedure based on Winslow's equation for unstructured grids. A detailed comparison showed this method to be efficient in transferring the moving boundary velocities to the internal nodes. These velocities were successfully applied in the flow solver.

Extension to moving composite grids : the single most important restriction to the basic sliding grid framework is the difficulty to adapt the grid density required by the moving boundary to that of the reference mesh into which it moves. This was resolved by transforming the moving mesh model from one of a body sliding through a reference grid to that of a boundary-fitted mesh sliding inside a reference mesh. In addition to the advantage of controlling grid concentration, the results obtained have shown the improved ability of the method to handle more complex motion-body shapes configurations.

Coupling to an ALE flow solver with GCL : the main step in the coupling of the moving mesh methodology was the extension of an existing finite volume solver for compressible flows to include the GCL to account for the mesh velocity. These procedures were applied to solve simple flows around moving bodies with varying geometric features (circular, square and wavy boundaries) to test the capability of the global algorithm. The simulations results proved adequate for the purpose of demonstrating the feasibility of the methodology and the present implementation.

7.2 Limitations of proposed work

In its present form, the proposed moving grid methodology is restricted as following. In each cases the possible remedy for future work is introduced.

Moving Boundaries in Absence of Collisions : the presented method is not still applicable for moving cases with zero proximity in boundaries (contact boundaries). This limitation comes from the fact that the method is inherently based on floating nodes with fixed connectivity. This weakness can be treated with a multi-blocked grid, where moving objects with zero proximity are placed inside a sub-domain and the sub-domain

is sliding through the main grid. Generally, one of the future developments can be the implementation of the sliding method combined with other mesh refinement and/or mesh adaptation approaches to get the highest efficiency of generated dynamic grid.

2D Configurations : in lots of engineering applications the simulation is necessary to be carried on in 3D spaces and the presented approach is not still well-prepared for these kind of situations. Then, the second shortcoming of the current work can be introduced as being in 2D spaces. One of the very first ideas for extending the method to 3D, is handling the sliding method only for a sphere in 3D as a moving sub-domain inside the main domain, and then setting the real 3D moving object inside this moving sphere. The stitching step can be like the 2D method as was explained in Chapter 4. This is another interesting area to be investigated in a future work.

Rigid Bodies : all case studies in the presented work were rigid bodies with no change in geometry during the motion. There are some engineering cases, e.g. in fluid structure interaction (FSI), where the geometry of moving object evolves during the time. For instance, this can be shown for a moving bubble with expansion/contraction during a time interval or for a moving fish with oscillating motions swimming in water. Although the presented methodology was introduced for rigid body, but with some manipulation it can be implemented for other cases as long as the connectivity remains fix. Since for the motions with non-rigid bodies the range of node displacements could be in a higher order, the two introduced smoothing methods that were used (i.e. Winslow and Laplace operators) can be change with a more sophisticated grid smoothing method like using Poisson equations with controllable forcing terms.

Explicit Time Marching Steps : in the proposed approach the path of the motion (trajectory) is predetermined and the moving object is restricted to move along this path. In some application this is a major shortcoming as the motion is solution dependant and the direction of the motion cannot be predicted at the beginning. Therefore, another interesting area for further research can be making the sliding procedure independent of the trajectory. One of the treatments can be refining the trajectory in each time step by aligning the leading edge to the direction of the motion.

Accurate Analyses : in addition to the listed items above, future developments required to bring the proposed moving grid methodology to a mature level is undertaking systematic validations and accuracy analyses for different flow regimes.

REFERENCES

En ligne : http://www.directindustry.fr

"Overset chemira method", http://celeritassimtech.com/?page_id=15={Accessed: 2015-09-16}.

C. Allen et T. Rendall, "Unified approach to cfd-csd interpolation and mesh motion using radial basis functions", dans 25th AIAA applied aerodynamics conference, 2007, pp. 25–28.

N. Aquelet, M. Souli, et L. Olovsson, "Euler–lagrange coupling with damping effects : Application to slamming problems", Computer methods in applied mechanics and engineering, vol. 195, no. 1, pp. 110–132, 2006.

S. Arabi, R. Camarero, et F. Guibault, "Unstructured meshes for large body motion using mapping operators", Mathematics and computers in simulation, vol. 106, pp. 26–43, 2014.

S. Arabi-Narehei, "Unstructured meshes for large body motion using mapping operators", dans PhD Thesis, Ecole Polytechnique de Montreal, 2012.

T. Baker, "Mesh deformation and reconstruction for time evolving domains", AIAA Paper, vol. 2535, p. 2001, 2001.

T. J. Baker, "Adaptive modification of time evolving meshes", Computer methods in applied mechanics and engineering, vol. 194, no. 48, pp. 4977–5001, 2005.

J. T. Batina, "Unsteady euler algorithm with unstructured dynamic mesh for complex-aircraft aerodynamic analysis", AIAA journal, vol. 29, no. 3, pp. 327–333, 1991.

J. Bau, H. Luo, R. Loehner, E. Goldberg, et A. Feldhun, "Application of unstructured moving body methodology to the simulation of fuel tank separation from an f-16 fighter", dans 35th Aerospace Sciences Meeting and Exhibit, AIAA Paper No. AIAA-1997-0166, Reno, NV, 1997.

T. Belytschko, J. M. Kennedy, et D. Schoeberle, "Quasi-eulerian finite element formulation for fluid-structure interaction", Journal of Pressure Vessel Technology, vol. 102, no. 1, pp. 62–69, 1980.

T. Belytschko, Y. Y. Lu, et L. Gu, "Element-free galerkin methods", International journal for numerical methods in engineering, vol. 37, no. 2, pp. 229–256, 1994.

F. Blom, “Considerations on the spring analogy”, International Journal for Numerical Methods in Fluids, vol. 32, no. 6, pp. 647–668, 2000.

J. Castillo, S. Steinberg, et P. Roache, “On the folding of numerically generated grids : use of a reference grid”, Communications in applied numerical methods, vol. 4, no. 4, pp. 471–481, 1988.

P. Chen et L. Hill, “A three-dimensional boundary element method for cfd/csd grid interfacing”, AIAA paper, pp. 99–1213, 1999.

V. Cristini, J. Bławzdziewicz, et M. Loewenberg, “An adaptive mesh algorithm for evolving surfaces : simulations of drop breakup and coalescence”, Journal of Computational Physics, vol. 168, no. 2, pp. 445–463, 2001.

I. Demirdžić et M. Perić, “Space conservation law in finite volume calculations of fluid flow”, International journal for numerical methods in fluids, vol. 8, no. 9, pp. 1037–1050, 1988.

J. Donéa, P. Fasoli-Stella, et S. Giuliani, “Lagrangian and eulerian finite element techniques for transient fluid-structure interaction problems”, dans Structural mechanics in reactor technology, 1977.

engines.polimi.it, “Grid motion for an engine valve”. En ligne : `http://www.engines.polimi.it/images/move/valveMotionGDI.gif`

M. Falsafioon, S. Arabi, R. Camarero, et F. Guibault, “Comparison of two mesh smoothing techniques for unstructured grids”, dans IOP Conference Series : Earth and Environmental Science, vol. 22, no. 2. IOP Publishing, 2014, p. 022020.

C. Farhat, C. Degand, B. Koobus, et M. Lesoinne, “Torsional springs for two-dimensional dynamic unstructured fluid meshes”, Computer methods in applied mechanics and engineering, vol. 163, no. 1-4, pp. 231–245, 1998.

A. Frisani et Y. A. Hassan, “On the immersed boundary method : finite element versus finite volume approach”, Computers and fluids, no. 121, 2015.

E. S. Gawlik et A. J. Lew, “High-order finite element methods for moving-boundary problems with prescribed boundary evolution”, Computer Methods in Applied Mechanics and Engineering, vol. 278, pp. 314–346, 2015.

W. J. Gordon et C. A. Hall, “Construction of curvilinear co-ordinate systems and applications to mesh generation”, International Journal for Numerical Methods in Engineering,

vol. 7, no. 4, pp. 461–477, 1973.

W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, et M.-G. Vallet, “Anisotropic mesh adaptation : Towards user-independent, mesh-independent and solver-independent cfd. part i : General principles”, International Journal for Numerical Methods in Fluids, vol. 32, no. 6, pp. 725–744, 2000.

O. Hassan, E. Probert, K. Morgan, et N. Weatherill, “Unsteady flow simulation using unstructured meshes”, Computer methods in applied mechanics and engineering, vol. 189, no. 4, pp. 1247–1275, 2000.

O. Hassan, K. Sørensen, K. Morgan, et N. Weatherill, “A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing”, International journal for numerical methods in fluids, vol. 53, no. 8, pp. 1243–1266, 2007.

B. Helenbrook, “Mesh deformation using the biharmonic operator”, International journal for numerical methods in engineering, vol. 56, no. 7, pp. 1007–1021, 2003.

T. J. Hughes, W. K. Liu, et T. K. Zimmermann, “Lagrangian-eulerian finite element formulation for incompressible viscous flows”, Computer methods in applied mechanics and engineering, vol. 29, no. 3, pp. 329–349, 1981.

S. R. Idelsohn et E. Onate, “To mesh or not to mesh. that is the question...”, Computer methods in applied mechanics and engineering, vol. 195, no. 37, pp. 4681–4696, 2006.

S. R. Idelsohn, M. A. Storti, et E. Oñate, “Lagrangian formulations to solve free surface incompressible inviscid fluid flows”, Computer Methods in Applied Mechanics and Engineering, vol. 191, no. 6, pp. 583–593, 2001.

A. Illinca, “Solution of three dimensional euler equations on unstructured moving grids”, dans PhD Thesis, Ecole Polytechnique de Montreal, 1996.

H. Jasak et Z. Tukovic, “Automatic mesh motion for the unstructured finite volume method”, Transactions of FAMENA, vol. 30, no. 2, pp. 1–20, 2006.

A. A. Johnson et T. E. Tezduyar, “Simulation of multiple spheres falling in a liquid-filled tube”, Computer Methods in Applied Mechanics and Engineering, vol. 134, no. 3, pp. 351–373, 1996.

——, “Advanced mesh generation and update methods for 3d flow simulations”, Computational Mechanics, vol. 23, no. 2, pp. 130–143, 1999.

K.-H. Kao, M.-S. Liou, et C.-Y. Chow, “Grid adaptation using chimera composite overlapping meshes”, AIAA, vol. 32, no. 5, 1994.

S. Karman et M. Sahasrabudhe, “Unstructured adaptive elliptic smoothing”, dans 45th AIAA Aerospace Sciences Meeting and Exhibit, AIAA Paper, vol. 559, 2007.

S. L. Karman Jr, “Virtual control volumes for two-dimensional unstructured elliptic smoothing”, dans Proceedings of the 19th International Meshing Roundtable. Springer, 2010, pp. 121–142.

T. Kempe et J. Fröhlich, “An improved immersed boundary method with direct forcing for the simulation of particle laden flows”, Journal of Computational Physics, vol. 231, no. 9, pp. 3663–3684, 2012.

D. B. Kholodar, S. A. Morton, et R. M. Cummings, “Deformation of unstructured viscous grids”, AIAA paper, vol. 926, p. 2005, 2005.

P. Knupp, “Winslow smoothing on two-dimensional unstructured meshes”, Engineering with Computers, vol. 15, no. 3, pp. 263–268, 1999.

P. Knupp et S. Steinberg, The fundamentals of grid generation. CRC, 1993.

S. Kwak et C. Pozrikidis, “Adaptive triangulation of evolving, closed, or open surfaces by the advancing-front method”, Journal of Computational Physics, vol. 145, no. 1, pp. 61–88, 1998.

M. Lesoinne et C. Farhat, “Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations”, Computer methods in applied mechanics and engineering, vol. 134, no. 1, pp. 71–90, 1996.

R. Löhner et C. Yang, “Improved ale mesh velocities for moving bodies”, Communications in numerical methods in engineering, no. 12, pp. 599–608, 1998.

R. Löhner, C. Yang, J. D. Baum, H. Luo, D. Pelessone, et C. M. Charman, “The numerical simulation of strongly unsteady flow with hundreds of moving bodies”, International Journal for Numerical Methods in Fluids, vol. 31, no. 1, pp. 113–120, 1999.

K. Masatsuka, I Do Like CFD, Vol. 1. Lulu. com, 2013, vol. 1.

A. Masud, M. Bhanabhagvanwala, et R. A. Khurram, “An adaptive mesh rezoning scheme for moving boundary flows and fluid–structure interaction”, Computers & fluids, vol. 36, no. 1, pp. 77–91, 2007.

R. L. Meakin, “On adaptive refinement and overset structured grids”, dans AIAA Paper No. AIAA-97-1858, 1997, pp. 236–249.

M. Murayama, K. Nakahashi, et K. Matsushima, “Unstructured dynamic mesh for large movement and deformation”, AIAA paper, vol. 122, p. 2002, 2002.

B. Nayroles, G. Touzot, et P. Villon, “Generalizing the finite element method : diffuse approximation and diffuse elements”, Computational mechanics, vol. 10, no. 5, pp. 307–318, 1992.

E. J. Nielsen et W. K. Anderson, “Aerodynamic design optimization on unstructured meshes using the navier-stokes equations”, AIAA journal, vol. 37, no. 11, pp. 1411–1419, 1999.

——, “Recent improvements in aerodynamic design optimization on unstructured meshes”, AIAA journal, vol. 40, no. 6, pp. 1155–1163, 2002.

B. Nkonga, “On the conservative and accurate cfd approximations for moving meshes and moving boundaries”, Computer methods in applied mechanics and engineering, vol. 190, no. 13, pp. 1801–1825, 2000.

E. Onate, S. Idelsohn, O. Zienkiewicz, et R. Taylor, “A finite point method in computational mechanics. applications to convective transport and fluid flow”, International journal for numerical methods in engineering, vol. 39, no. 22, pp. 3839–3866, 1996.

E. Onate, S. Idelsohn, O. Zienkiewicz, R. Taylor, et C. Sacco, “A stabilized finite point method for analysis of fluid mechanics problems”, Computer Methods in Applied Mechanics and Engineering, vol. 139, no. 1, pp. 315–346, 1996.

C. S. Peskin, “Flow patterns around heart valves : a numerical method”, Journal of computational physics, vol. 10, no. 2, pp. 252–271, 1972.

A. Pigeon, “Développement d’une méthode d’accélération par grilles virtuelles récursives pour l’assemblage de maillages chimères”, Mémoire de maîtrise, École Polytechnique de Montréal, 2015.

R. Rangarajan et A. J. Lew, “Universal meshes : A new paradigm for computing with nonconforming triangulations”, arXiv preprint arXiv :1201.4903, 2012.

M. Reggio, J. Trepanier, H. Zhang, et R. Camarero, “Numerical simulation of the gas flow in a circuit-breaker”, International journal for numerical methods in engineering, vol. 34, no. 2, pp. 607–618, 1992.

T. Rendall et C. Allen, “Unified fluid–structure interpolation and mesh motion using radial basis functions”, International Journal for Numerical Methods in Engineering, vol. 74, no. 10, pp. 1519–1559, 2008.

——, “Efficient mesh motion using radial basis functions with data reduction algorithms”, Journal of Computational Physics, vol. 228, no. 17, pp. 6231–6249, 2009.

P. L. Roe, “Approximate riemann solvers, parameter vectors, and difference schemes”, Journal of computational physics, vol. 43, no. 2, pp. 357–372, 1981.

M. Sahasrabudhe, Unstructured mesh generation and manipulation based on elliptic smoothing and optimization, 2008.

P. Saksono, W. Dettmer, et D. Perić, “An adaptive remeshing strategy for flows with moving boundaries and fluid–structure interaction”, International Journal for Numerical Methods in Engineering, vol. 71, no. 9, pp. 1009–1050, 2007.

D. Schillinger, L. Dede, M. A. Scott, J. A. Evans, M. J. Borden, E. Rank, et T. J. Hughes, “An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of nurbs, immersed boundary methods, and t-spline cad surfaces”, Computer Methods in Applied Mechanics and Engineering, vol. 249, pp. 116–150, 2012.

D. M. D. I. R. Schneiders, W. Oberschelp, R. Kopp, et M. Becker, “New and effective remeshing scheme for the simulation of metal forming processes”, Engineering with computers, vol. 8, no. 3, pp. 163–176, 1992.

E. F. Sheta, H. Yang, et S. D. Habchi, “Solid brick analogy for automatic grid deformation for fluid–structure interaction”, dans 36th AIAA Fluid Dynamics Conference and Exhibit, AIAA Paper No. AIAA-2006-3219, San Francisco, CA, 2006.

R. L. Sorenson, “A computer program to generate two-dimensional grids about airfoils and other shapes by the use of poisson’s equation”, NASA Technical Memorandum, vol. 81198, pp. 1–58, 1980.

F. Sotiropoulos et X. Yang, “Immersed boundary methods for simulating fluid–structure interaction”, Progress in Aerospace Sciences, vol. 65, pp. 1–21, 2014.

J. L. Steger, F. Dougherty, et J. A. Benek, “A chimera grid scheme”, dans Advances in Grid Generation, vol. 5. ASME, 1983, pp. 59–69.

J. F. Thompson, B. K. Soni, et N. P. Weatherill, Handbook of grid generation. CRC press, 1998.

J. Trépanier, M. Reggio, M. Paraschivoiu, et R. Camarero, “Unsteady euler solutions for arbitrarily moving bodies and boundaries”, AIAA journal, vol. 31, no. 10, pp. 1869–1876, 1993.

V. Venkatakrishnan et D. Mavriplis, “Implicit method for the computation of unsteady flows on unstructured grids”, Journal of Computational Physics, vol. 127, no. 2, pp. 380–397, 1996.

X. Wang, L.T.Zhang, et W. Liu, “On computational issues of immersed finite element methods”, J. Comput. Phys., vol. 228, no. 2, pp. 2535–2551, 2009.

Z. J. Wang et V. Parthasarathy, “A fully automated chimera methodology for multiple moving body problems”, International Journal for Numerical Methods in Fluids, vol. 33, pp. 919–938, 2000.

A. M. Winslow, “Numerical solution of the quasilinear poisson equation in a nonuniform triangle mesh”, Journal of computational physics, vol. 1, no. 2, pp. 149–172, 1966.

Z. Yang et D. Mavriplis, “Unstructured dynamic meshes with higher-order time integration schemes for the unsteady navier–stokes equations”, AIAA paper, vol. 1222, 2005.

——, “Higher-order time-integration schemes for aeroelastic applications on unstructured meshes”, AIAA journal, vol. 45, no. 1, p. 138, 2007.

X. Zheng, J. Lowengrub, A. Anderson, et V. Cristini, “Adaptive unstructured volume remeshing–ii : Application to two-and three-dimensional level-set simulations of multiphase flow”, Journal of Computational Physics, vol. 208, no. 2, pp. 626–650, 2005.

X. Zhou et S. Li, “A new mesh deformation method based on disk relaxation algorithm with pre-displacement and post-smoothing”, Journal of Computational Physics, vol. 235, pp. 199–215, 2013.