UNIVERSITÉ DE MONTRÉAL

UNSTRUCTURED MESHES FOR LARGE RIGID BODY MOTIONS USING MAPPING OPERATORS

SINA ARABI NAREHEI
DÉPARTEMENT DE GÉNIE MÉCANIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE MÉCANIQUE)
MARS 2012

© Sina Arabi Narehei, 2012.

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

UNSTRUCTURED MESHES FOR LARGE RIGID BODY MOTIONS USING MAPPING OPERATORS

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

M. LAURENDEAU Eric, Ph.D., président
M. CAMARERO Ricardo, Ph.D., membre et directeur de recherche
M. GUIBAULT François, Ph.D., membre et codirecteur de recherche
M. LAFOREST Marc, Ph.D., membre
M. PARASCHIVOIU Marius, Ph.D., membre

Dedicated to my parents

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisors Professor Ricardo Camarero and Professor François Guibault for the continuous support of my Ph.D study and research, for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis.

My sincere thanks also goes to Professor Chris Allen from University of Bristol who has taught me the RBF method during his stay in Montreal.

I warmly thank Professor Dominique Pelletier, for his valuable advices and helps. His discussions around my work have been very helpful for this study. Also, I would like to thank Dr. Marie-Gabrielle Vallet for her advices and comments.

My time in Montreal was made enjoyable in large part due to the many friends and groups that became a part of my life. I am grateful for time spent with all of them and our memorable moments.

Last but not the least, I would like to express my exceptional gratitude to my brother, Shahram and his family Shabnam, Parsa and Arya for their continuous and loving support.

RÉSUMÉ

Cette thèse propose une approche originale pour le contrôle des maillages autours d'objets rigides en mouvement. L'approche proposée permet de maintenir fixe la topologie du maillage, et ainsi d'éliminer le recours à un processus d'interpolation des solutions entre les pas de temps lors de simulations en régime transitoire. Afin de simplifier le traitement du mouvement des objets, leur évolution est décrite dans un espace de calcul, qui est par la suite transformé vers l'espace physique grâce à des opérateurs différentiels. Deux types d'opérateurs différentiels ont été étudiés, les premiers inspirés de fonctionnelles de forme des éléments (Longueur, Aire et Orthogonalité),
et les seconds des équations de Winslow. L'une des contributions principales de cette étude est l'extension des équations d'Euler-Lagrange aux fonctionnelles de forme, ainsi qu'à leurs combinaisons, et l'application de ces fonctionnelles au traitement de maillages non-structurés. Deux techniques distinctes de discrétisation de ces équations aux dérivées partielles ont été étudiées. La première technique est basée sur un schéma de différences finies à neuf points, et la seconde sur un schéma de volumes finis utilisant une linéarisation des opérateurs.
Une seconde contribution a consisté à introduire la notion de glissement des nœuds du maillage sur les frontières des objets en mouvement. En intégrant les techniques de glissement des nœuds, gérées dans l'espace de calcul, et les techniques de transformation de l'espace de calcul vers l'espace physique, une approche robuste de contrôle des maillages pour de très grands déplacements des objets a pu être mise au point.
Cette approche, combinée à une seconde méthode de gestion du mouvement dans l'espace physique utilisant les fonctions à bases radiales, a permis de traiter des configurations d'objets en mouvement le long de trajectoires complexes. La méthodologie globale a ainsi pu être utilisée pour traiter des configurations représentatives d'applications en ingénierie.

ABSTRACT

The main objective of this thesis was to generate unstructured meshes with fixed connectivity for large rigid body motion. The proposed approach consists in generating a mesh in computational space for a generic configuration of the moving body. The management of body and mesh motion is then carried out in computational space using a sliding mesh paradigm. Afterwards, the mesh in physical space is obtained through PDE mapping operators. Two different mapping operators based on functionals and Winslow equations have been investigated to recover the physical space by the computational mesh.

One of the main contributions of this study is extending the Euler-Lagrange equations of Length, Area, Orthogonality functionals and their combinations, to unstructured grid technologies. Two new discritization techniques are implemented, validated and compared for performing different mapping operators on unstructured grids. The first approach used a 9-point cartesian stencil inside each patch of the computational mesh and discritizes the mapping operators on that using conventional finite difference schemes. The second approach used finite volume discritization technique by linearizing the system of mapping equations. Finally, the Radiad basis Functions interpolation technique can be used as a secondary mesh motion technique and after the mesh sliding procedure. Combination of these two techniques allow us to handle more complex trajectories of the boundaries in physical space.

The overall methodology is applied to complex geometric configurations representative of engineering applications.

TABLE OF CONTENTS

DEDICATION iii

ACKNOWLEDGMENTS iv

RÉSUMÉ v

ABSTRACT vi

TABLE OF CONTENTS vii

LIST OF FIGURES x

LIST OF TABLES xv

CHAPTER 1 INTRODUCTION 1

1.1 Problem formulation 3

1.1.1 Mesh motion by topological change 4

1.1.2 Mesh motion by fixed topology 5

1.2 Proposed approach 7

1.3 Objectives 8

1.4 Thesis outline 10

CHAPTER 2 MESH MOTION IN PHYSICAL DOMAIN 12

2.1 Mesh deformation based on fixed connectivity 12

2.1.1 Laplace equation 12

2.1.2 Linear elasticity 14

2.1.3 Radial basis functions 17

2.2 Mesh movement by changing connectivity 21

2.3 Discussion 24

CHAPTER 3 MESH MOVEMENT USING SLIDING CELLS 26

3.1 A flow analogy for mesh motion 26

3.2 Application of the method 30

3.2.1 Restriction . . . 35
3.3 Evaluation of the method . . . 37

CHAPTER 4 MAPPING DOMAINS . . . 39
4.1 *"A mesh to make a mesh"* . . . 39
4.2 Mapping Models . . . 41
4.2.1 The Winslow operator . . . 41
4.2.2 Mesh mapping by functionals . . . 42
4.3 Ellipticity study of mapping operators . . . 44
4.4 Numerical discretization . . . 46
4.4.1 Finite difference scheme . . . 47
4.4.2 Finite volume scheme . . . 49
4.5 Validation of the mapping methodology . . . 52
4.5.1 Validation of the finite difference scheme . . . 53
4.5.2 Comparison of finite volume and finite difference schemes . . . 54
4.5.3 Validation and comparison of different mapping operators . . . 56
4.6 Mesh smoothness . . . 60
4.7 Limitations . . . 68
4.8 Discussion . . . 75

CHAPTER 5 APPLICATION TO MESH MOTION . . . 76
5.1 Generic boundary in the computational domain . . . 76
5.2 Translational motion . . . 78
5.2.1 Sliding a circle inside a cavity . . . 78
5.2.2 Sliding a circle past obstacles . . . 84
5.3 Rotary motion . . . 92
5.3.1 Rotation of a NACA0012 airfoil . . . 92
5.3.2 Four-petal rose rotation . . . 93
5.3.3 Relative rotary motion . . . 94
5.4 Combination of translational and rotational motion . . . 94
5.4.1 Sliding a NACA0012 airfoil on a sinusoidal trajectory . . . 95
5.4.2 Cobra-like motion of a NACA0012 airfoil . . . 95

CHAPTER 6 CONCLUSIONS AND FUTURE WORK 109
6.1 Conclusions 109
6.2 Future work 111

REFERENCES 111

LIST OF FIGURES

Figure 1.1 Examples of unsteady flow requiring moving meshes 2
Figure 1.2 Classification of Different moving mesh Methodologies 3
Figure 1.3 Simple deformation of an element due to its two nodes' movement 9
Figure 2.1 Comparison of mesh displacement using boundary displacement and velocity, computed using Laplace equation 15
Figure 2.2 Propagated mesh velocity vector on the interior nodes 16
Figure 2.3 Incremental rotation of a NACA0012 airfoil in 360° using the linear elasticity model [figure courtesy of professor R.P. Dwight TU Delft, Dwight (2006)] . . 17
Figure 2.4 Initial mesh around NACA0012 and mesh quality 20
Figure 2.5 Deformed mesh around a rotated NACA0012 airfoil at four angular position using Radial Basis Functions interpolation method 20
Figure 2.6 Mesh quality at four angular positions for the rotated NACA0012 airfoil . . . 21
Figure 2.7 Initial mesh around a four-petal rose and mesh quality 22
Figure 2.8 Deformed mesh around a rotated four-petal rose at four angular position using Radial Basis Functions interpolation method 22
Figure 2.9 Mesh quality at four angular positions for the rotated four-petal rose 23
Figure 2.10 Swapping the flattened mesh to avoid mesh breakdown 24
Figure 2.11 Moving and deforming boundary in domain based on swapping edges and smoothing nodes 25
Figure 3.1 Sliding cells on a moving solid boundary 27
Figure 3.2 Generated mesh and direction of motion of the cells past on a circle 28
Figure 3.3 Treatment of the leading and trailing edges 29
Figure 3.4 Sliding an airfoil through the cells 31
Figure 3.5 Created flattened cells ahead of a moving circle in $-X$ direction 32
Figure 3.6 ϕ field around a NACA0012 33
Figure 3.7 $\nabla\phi/|\nabla\phi|$ vectors around a NACA0012 33
Figure 3.8 The mesh (a) and ϕ (b) at $t = 1$ 34
Figure 3.9 The mesh (a) and ϕ (b) at $t = 50$ 34
Figure 3.10 The mesh (a) and ϕ (b) at $t = 100$ 34
Figure 3.11 Ideal configuration for cell parting ahead of leading edge 35

Figure 3.12 Invalid cells after separation along a single edge 36
Figure 3.13 Invalid cells after separation along a misaligned edge 36
Figure 3.14 Swapping applied only on edge attached to the L.E. 37
Figure 4.1 Mapping of a computational domain (ξ, η) to physical space (x, y) 40
Figure 4.2 Structured and unstructured discretization stencils 47
Figure 4.3 Proposed 9 point stencil for the finite difference discretization of the mapping operation for unstructured mesh 48
Figure 4.4 Missing information by cell(M,P,N) on the 9 point stencil 49
Figure 4.5 Computational mesh and a control volume 50
Figure 4.6 Augmented stencil used in Karman (2010) for cross derivative terms 51
Figure 4.7 Solving Laplace's equation to transfer physical mesh to computational domain using a finite volume discretization technique, and solving Winslow equation on the computational mesh using the modified finite difference scheme to recover physical space by the computational mesh. 53
Figure 4.8 Mapping from physical to computational space 54
Figure 4.9 Final mesh in physical domain 55
Figure 4.10 The error value at each node indicates the difference between the original and the resulting mesh obtained by finite difference scheme 55
Figure 4.11 Mapping a straight line with coincided nodes to a circle using Winslow equations and finite difference scheme 56
Figure 4.12 Mapping of a slit in computational domain to a spike in physical space using Winslow equations 57
Figure 4.13 A straight line with coinciding nodes mapped to a circle using the Winslow operator 58
Figure 4.14 Comparison of the two mapping operators 59
Figure 4.15 Comparison of the mesh quality for the two mapping operators using Minimum angle distribution 60
Figure 4.16 Comparison of individual functionals for a spike geometry 61
Figure 4.17 Comparison of combined functionals for a spike geometry 62
Figure 4.18 Comparison of mesh quality and mesh smoothness by 'A' functional where $SF = 0.746$ 64

Figure 4.19 Comparison of mesh quality and mesh smoothness by 'O' functional where $SF = 0.739$ 65
Figure 4.20 Comparison of mesh quality and mesh smoothness by 'AO' functional where $SF = 0.817$ 66
Figure 4.21 Comparison of mesh quality and mesh smoothness by Winslow operators where $SF = 0.824$ 67
Figure 4.22 Comparison of individual functionals for a four-petal rose geometry 69
Figure 4.23 Comparison of combined functionals for a four-petal rose geometry 70
Figure 4.24 (a) Computational mesh, (b) bump and (c) spike geometries 72
Figure 4.25 Comparison of individual and combined functionals around an extreme bump 73
Figure 4.26 Comparison of individual and combined functionals around an extreme spike 74
Figure 5.1 Mapping of a generic configuration for translation motion 77
Figure 5.2 Mapping of a generic configuration for rotational motion. 77
Figure 5.3 Arrangement of the nodes on the slit 79
Figure 5.4 Sliding mesh nodes on the slit 79
Figure 5.5 Corresponding mesh node on the boundary 79
Figure 5.6 Computational and corresponding physical mesh at first, intermediate and final steps 80
Figure 5.7 Iso-η lines in (a) computational and (b) physical domains by Winslow operators 81
Figure 5.8 Mesh and iso-η lines in physical domain by A functional 81
Figure 5.9 Mesh and iso-η lines in physical domain by O functional 82
Figure 5.10 Mesh and iso-η lines in physical domain by AO functional 82
Figure 5.11 Invalid mesh in physical domain by L functional 83
Figure 5.12 Comparison of mesh smoothness around the moving circle at the intermediate step for different mapping operators 83
Figure 5.13 Comparison of mesh quality around the moving circle at the intermediate step for different mapping operators 85
Figure 5.14 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=1 86
Figure 5.15 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=25 87

Figure 5.16 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=42 88
Figure 5.17 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=105 89
Figure 5.18 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=128 90
Figure 5.19 Modelling the boundary proximity using the Winslow operator at four different time steps 91
Figure 5.20 Rotating physical boundary and sliding computational boundary over the mesh points 93
Figure 5.21 Updating the node numbering of the cells attached to the boundary as the result of sliding the boundary for three consecutive time steps 97
Figure 5.22 Rotating and sliding the cells around an airfoil in the physical domain for six different positions 98
Figure 5.23 Sliding the cells around the boundary in computational and physical domain at three different steps 99
Figure 5.24 Boundaries in (a) computational and (b) physical spaces 100
Figure 5.25 Sliding the cells around the boundaries in computational and physical domain at three different steps 101
Figure 5.26 SF values for 360^o rotary motion of two four-petal roses 102
Figure 5.27 (a) Computational, (b) parametric and (c) physical spaces used for combination of translational and rotational motion 102
Figure 5.28 Sinusoidal trajectory in physical domain 102
Figure 5.29 Linear motion of (a) the slit in the computatioal space and (b) the NACA0012 airfoil in the parametric space 103
Figure 5.30 Linear motion of a NACA0012 airfoil in the parametric space 104
Figure 5.31 Sinusoidal motion of a NACA0012 airfoil in the physical space obtained by sliding mesh and RBF method as a secondary mesh motion technique 105
Figure 5.32 SF values for sinusoidal motion of a NACA0012 airfoil 106
Figure 5.33 Cobra-like trajectory for a NACA0012 airfoil in physical domain 106

Figure 5.34 Cobra-like motion of a NACA0012 airfoil in the physical space using sliding cells and RBF method as a secondary mesh motion technique 107
Figure 5.35 SF values for Cobra-like motion of a NACA0012 airfoil 108

LIST OF TABLES

Table 2.1 Basis functions for $\| x \| < 1$ 19

Table 4.1 Comparison of global Smoothness Factor, "SF", for the spike geometry, based on different mapping operators 68

Table 4.2 Comparison of global Smoothness Factor, "SF", for the four-petal geometry, based on different mapping operators 68

Table 5.1 Values of SF for the sliding circle at intermediate step, case in Fig. 5.6 84

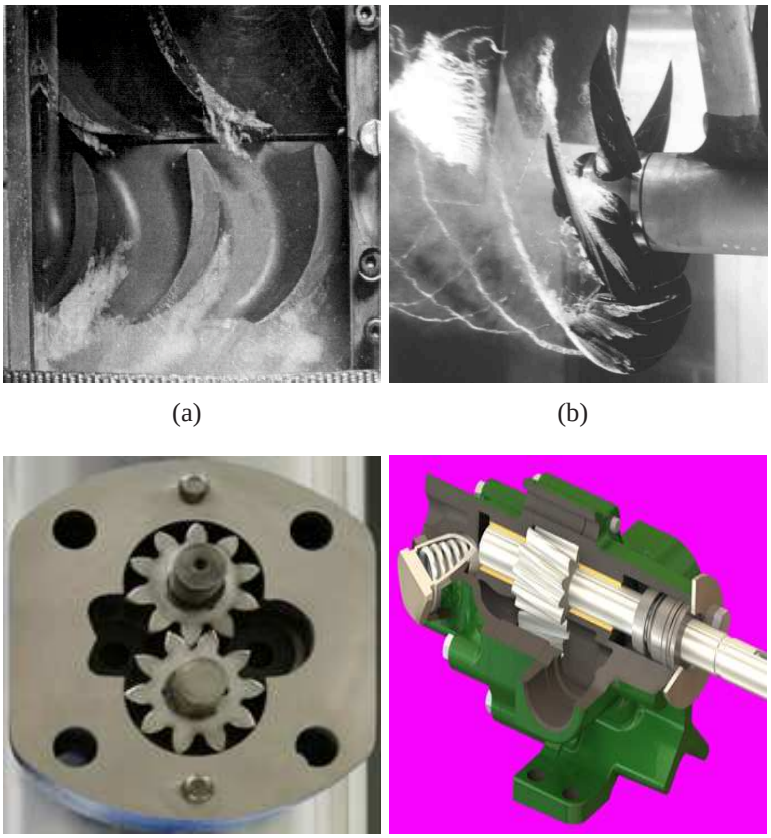
CHAPTER 1

INTRODUCTION

Computational fluid dynamics (CFD), through the integration of innovative numerical techniques, provides an attractive methodology for the simulation of flow-based applications around complex geometries. Within its very short life span, enormous advances in both solvers and geometric modelling have been achieved, turning these into reliable tools and brought them within the reach of the research and application engineer.

Recently, solvers for treating high Reynolds number flows using unstructured grids for both viscous and inviscid regions have been proposed by Hassan et al. (1998), Lin et al. (2001), Yang & Mavriplis (2005) and Hassan et al. (2007). Extensions to unsteady flows simulations, based on the full Reynolds averaged Navier-Stokes equations using different types of turbulence modelling and grids have been proposed successfully [Rhee (2005, 2009)]. These achievements depend, entirely on the grid generation procedure which is critically important in providing a link between the physics and the geometry. This is specially the case for unsteady flows including moving boundaries.

Such unsteady phenomena occur, for instance, in turbomachinary problems which require considerable computational effort with regards to both geometry, discretization and physics modelling. As illustrated in Fig. 1.1(a), the flow in these problems is inherently unsteady due to the relative motion between the different components of the machine. Another example, shown in Fig. 1.1(b), is a complex flow around a screw propeller, commonly used in marine industries. The study of the flow around these configurations is characterized by a boundary motion of large amplitude. A more complex situation occurs in the case of two boundaries that are in relative motion, as illustrated in Figs. 1.1(c) and (d) for the flow in a gear pump. In addition to the complicated geometry, the effects of boundary proximity and large relative motion contribute to the difficulties of these problems. Furthermore, the flow is fully turbulent, highly three-dimensional and spatially non-uniform. It is thus a veritable challenge to simulate accurately such complex flow fields. In order to capture the fluid features in such unsteady problems, the mesh motion should be directly coupled and adapted to the flow field simulation and its unsteadiness. Hence, the goal of mesh adaptation is to increase the accuracy of numerical calculations subject within the constraint of computational costs and an acceptable error.



(a) (b)

(c) (d)

Figure 1.1 Examples of unsteady flow requiring moving meshes

To simulate such problems, the efficient and reliable generation of time evolving meshes is a preliminary requirement. This is expressed in terms, on one hand, of a valid mesh for smoothness and quality, and on the other hand, of meshing tools capable of successful treatment of prescribed body motions. These two factors provide a continuing challenge for transient mesh generation methodology.

Unstructured mesh generation techniques are capable to meet certain element properties such as size control and flexibility, in addition to providing suitable environments for automation. Unlike their structured counterpart, unstructured meshes can be modified locally by insertion, removal, refinement and coarsening of elements.

State of the art mesh motion methodologies applicable to unsteady problems can be divided into two major categories, as illustrated in Fig. 1.2, namely, one group which maintains the mesh topology whereas the other is based on changing topology. These two groups are briefly reviewed in the next section by considering their advantages and limitations.

1.1 Problem formulation

For configurations involving moving boundaries, specially with large body motions such as turbomachinary problems, mixer blades and control surface deflection problems, a robust mesh movement technique is a requisite.

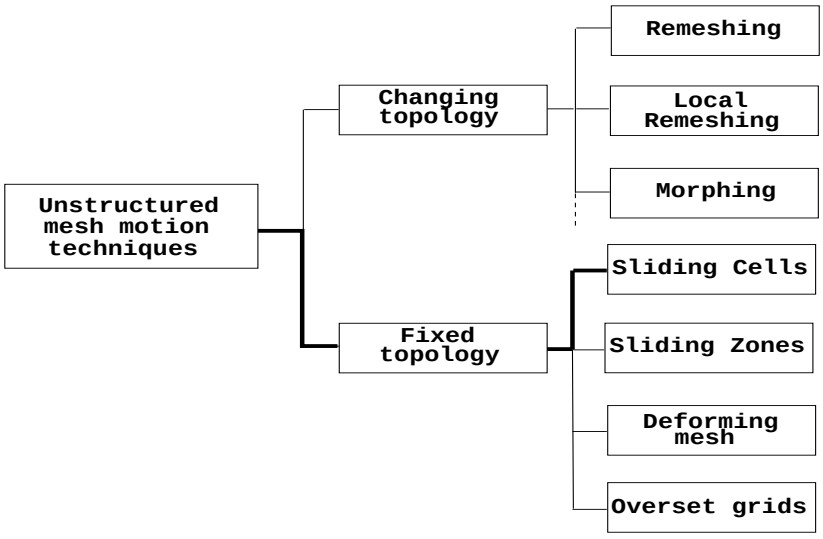


Figure 1.2 Classification of Different moving mesh Methodologies

The most demanding characteristic of an unsteady flow with regards to moving grid generation is when large amplitude of the relative motion of the boundaries are present. For instance, the simulating of an oscillatory airfoil can be handled successfully using the spring analogy technique for small displacement about a reference position, but this will fail for large amplitudes.
Modelling viscous effects to capture flow features near the boundaries requires very high aspect ratio cells, where the spacing of the first layer from the boundary can be 10^{-6} times the domain characteristic length. Thus, even with a very small motion, the boundary can traverse many layers of nearby cells, potentially leading to invalid meshes. The problem becomes more difficult for body motion in close proximity or in contact.
Considerable efforts have been addressed at these types of problems and several mesh movement strategies have been successfully demonstrated in the literature. In this section, these strategies are categorized in two main groups, based on changing and fixed topology of the mesh, as shown in Fig. 1.2. These are reviewed in terms of their applicability to large relative motion, and assessed with respect to robustness and efficiency.

1.1.1 Mesh motion by topological change

The early attempts in dynamic grid generation are essentially remeshing techniques, where the entire grid is regenerated based on the new position of the boundaries as presented in Anderson et al. (2005), Kwak & Pozrikidis (1998), Saksono et al. (2007), Schneider et al. (1992) and Cristini et al. (2001). This approach has been applied for adaptive flow simulations as well as for the simulation of flow with moving bodies. This can produce meshes of very high quality if the defined size function is well behaved, for example by equidistributing the error across the domain [Persson (2005)]. However, it is very expensive computationally, as in addition to the actual remeshing, the technique requires the interpolation of the solution at each time step.

A major improvement of the efficiency of this approach is to apply remeshing locally as determined by a mesh quality indicator, [Hassan et al. (2000)]. Based on this indicator, elements are removed, resulting in one or more voids in the mesh which are then remeshed according to the required distribution of mesh parameters provided by the error indicating process and merged into the global mesh, [Hassan et al. (1998)]. Although applied locally, compared to the complete remeshing, this method remains computationally expensive for transient problems. 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 transferring data from one mesh to another [Hassan et al. (2007)]. In addition, specially in 3D cases, the generated void can be nonconvex, 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 voids has to be remeshed to be prepared for applying the volume meshers. This procedure increases the complexity of the method considerably.

Another mesh adaptation method, called morphing, applies local edge collapse operations for mesh coarsening, and incremental point insertion algorithms for mesh refinement. These procedures always require and operate on a valid mesh [Baker (2001, 2003)]. For moving mesh applications, these algorithms can be integrated with the mesh movement procedure to form an adaptation algorithm that will modify the mesh of a domain whose shape is evolving in time. If inverted elements are created after moving the boundary, the motion step is halved and the process is continued. In general, a given movement step of the boundary can be broken up into such smaller steps. If the situation persists, the step can be further refined. Although the method is very flexible, its robustness depends heavily on maintaining mesh quality during each adaptation cycle.

1.1.2 Mesh motion by fixed topology

Another approach to mesh motion is to adjust the computational or physical mesh to the moving boundaries by the movement of all internal nodes based on the prescribed boundary motion, without modifying the mesh connectivity. Such mesh motion techniques where the connectivity remains fixed are called mesh deformation.

Deforming mesh algorithms have been presented in the literature, with various approaches according to the amplitude of the body motion. A widely used method has been presented by Batina (1991) which considers the mesh as a network of linear springs and solves the static equilibrium equations for this network to determine the new location of the grid points. A disadvantage of this approach is that the grid smoothness and regularity are lost when the grid is subjected to large motion.
This has been improved in Farhat et al. (1998) by the additional torsional springs for controlling the arbitrary motion of grid points. 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. However, despite these improvements, this approach proved to lack of robustness particularly for large deformations.

Another promising method is based on the Radial Basis Functions interpolation presented in Rendall & Allen (2009) which can be applied to mesh motion while preserving the cells' connectivity. This is an interpolation technique where the displacements of boundary nodes are propagated onto the interior nodes. A small system of equations, involving only the boundary nodes, has to be solved and no mesh connectivity information is needed. The method can handle large mesh deformations caused by translations, rotations and deformations, for both 2D and 3D meshes. This can be compared to PDE-based approaches in terms of mesh quality and efficiency for motions where the surface deformations are smooth, such as in fluid-structure interaction problems. This has been applied successfully for large relative motions in engineering applications but since the cells remain attached to the boundary and the connectivity matrix remain fixed, it is not suitable for periodic rotary objects like modelling the mixer blades or propellers.

PDE operators, such as Laplace equation with various diffusivity coefficients, have been used as a mechanism to generate and to smooth meshes. 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 [Illinca et al. (1995), Trépanier et al. (1993) and Zhang et al. (1993)], thus insuring valid grids. While the behavior of the Laplace equation is similar to that of the spring analogy, Poisson's equation with forcing functions provides a better control on the mesh deformation procedure. These control 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, the secret of each "good" system of Poisson equations is selecting appropriate forcing terms, which is thus equivalent to constructing valid grids in mesh deformation technique.

Notably, in an effort to simultaneously control both, edge lengths, which can be achieved by Laplace's equation, and the normal mesh spacing in moving boundary problems, Helenbrook (2001) proposed the use of biharmonic equations. In this technique, not only mesh spacings will be continuous at interfacial or periodic boundaries, but it also allows the simulation of problems with greater boundary deformation than second-order methods or Laplace equations. While this method is valuable for both

free-surface and interfacial flow calculations, it still has limitations for the large linear and rotary motions in unsteady flow problems.

Another attempt to solve large mesh deformation has been proposed by Yang & 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 & 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 enhanced 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, specially around high curvature regions or sharp corner points of boundaries. This is due to the type of the system of equations, which is elliptic and will be further discussed in Chapter 4.

Another technique is overset grids [Benek et al. (1986)] that simplify the mesh management by superposition of the static and moving parts of the grids at the expense of the cost of interpolation.

In spite of the considerable advances achieved in the area of unstructured mesh movement, there remain several critical issues relating to robustness and efficiency of these methods. This thesis presents a contribution to this generic problem with the specific goal to extend the range of applicability of relative large body movement and boundary proximity.

1.2 Proposed approach

The ability to move objects in a physical domain gives the possibility of simulating unsteady engineering problems. To do that, the generation of an initial mesh with respect to the desired resolution of the initial solution is required. The solution is advanced in time and, at each time step, the coordinates of the points are updated according to the prescribed or computed movement of the boundaries. It is clear that, without using adaptive and mesh motion techniques, the mesh becomes more and more distorted and eventually invalid.

The methods described so far, all share one major characteristic, which is that the mesh motion is carried out in physical space. Generally, each method satisfies a particular set of requirements at the expense of their important capabilities. For example, large motions can be handled by local remeshing techniques but present difficulty with accuracy and complexity. Similarly, while the spring analogy is very efficient and easy to apply, it fails for large motions. Furthermore, despite their significant benefits, the Radial Basis Functions interpolation, linear elasticity techniques and biharmonic equations are still not capable of handling large linear or periodic rotary motions encountered in the engineering applications. This is due to the constraint of the attachment of cells to the boundaries and the difficulty of maintaining mesh topology as the motion evolves.

A new approach is presented in this thesis, based on a flow analogy, where the grid cells are advected past moving boundaries by a fictitious potential-like fluid flow. In this approach, the cells are allowed to slide on the boundaries in order to release the constraints present in the conventional mesh deformation techniques. In addition, the cells connectivities will be maintained except at specific points on the boundary. Despite this method's capability to handle large motions and even contact and separating boundaries, it requires special treatment of grid cells on the boundaries. This is especially the case for bluff bodies where additional forcing terms are required as to maintain the mesh validity and quality explained in the following chapters.
The way around this difficulty and instead of dealing with forcing terms, the sliding procedure can be managed in computational space, where the boundaries are simplified and the cells remain isotropic for all the evolving steps. Then, the computational mesh is mapped to the physical domain to achieve the respective physical mesh.

1.3 Objectives

In the previous sections, several moving mesh techniques have been reviewed for large relative motions. From this analysis many of the difficulties encountered can be related to the fact that the mesh remains attached to the moving boundaries, resulting in invalid meshes for large deformation. Another difficulty arises from the management of the grid motion in physical space.
In the present work, it is proposed specifically, to follow and develop the "Fixed topology" branch shown in Fig. 1.2 in order to achieve the goal of modeling mesh movement generated by large rigid

body motions, while lifting the constraint that mesh nodes remain attached to moving boundaries. The main advantage of this approach is to preserve the mesh connectivity as time evolves, making it well suited for Arbitrary Lagrangian Eulerian (ALE) flow solvers. By its very definition, the ALE approach combines both the Lagrangian and Eulerian reference frames and allows for a flexible, moving grid. This is helpful in problems with large deformation of boundaries where the grid tracks the fluid or boundary. In addition, an often overlooked issue in moving grids is the discretization of the Geometric Conservation Law (GCL) presented in Farhat & Lesoinne (1996). These consist in two equations that 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). These requirements for the time dependent meshes in a finite volume method have been presented in Zhang et al. (1993), Trépanier et al. (1993) and are implicitly verified in the present approach as the grid connectivity is fixed. For example, the continuity equation on a moving grid is

$$\frac{\partial V}{\partial t} - \oint_{\partial C} \vec{w} \cdot \vec{ds} = 0 \tag{1.1}$$

Where $\partial V/\partial t$ is the rate of change of the cell's volume and ω represents the velocity of control volume surfaces. If the above relation is satisfied, the volume would be conserved for all the control volumes. Consider Fig. 1.3, in which grid points are moving. This figure reveals how the size of subcontrol volume 2qop changes due to the motion of nodes 1 and 3. Therefore, to satisfy VCL on the shown triangle, the normal component of velocities of subsurfaces a and b (i.e. $(\overrightarrow{w_n})_a$ and $(\overrightarrow{w_n})_b$) are evaluated as follows

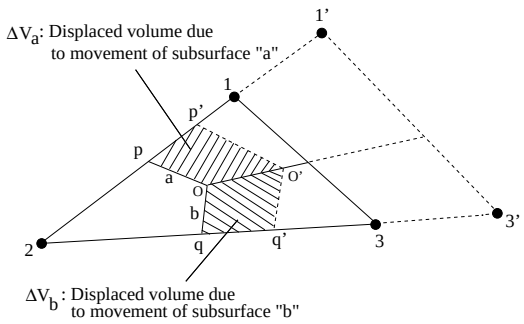


Figure 1.3 Simple deformation of an element due to its two nodes' movement

$$(\overrightarrow{w_n})_a = \frac{\Delta V_a}{\Delta t} \cdot s_a \quad , \quad (\overrightarrow{w_n})_b = \frac{\Delta V_b}{\Delta t} \cdot s_b \tag{1.2}$$

where s_a and s_b are surfaces of a and b, and ΔV_a and ΔV_b are the swept volumes by their corresponding surfaces. Based on this definition of subsurface velocities, the VCL law would be satisfied.

The strategy to address these issues is based on the decoupling of the grid management and domain geometry requirements by;

1. sliding cells on the boundary,
2. managing grid motion in computational space, which decreases difficulties in handling physical domain features, especially in regions where boundary curvature varies rapidly,
3. achieving geometric conformity through a mapping procedure from a generic grid in computational space to physical space.

1.4 Thesis outline

The subsequent chapters of this desertation deal with the approaches that were developed for the purpose of unstructured mesh motion with large amplitudes and fixed connectivity of cells.
First of all, the mesh deformation and mesh motion techniques in physical domain are studied and developed in Chapter 2. This is to understand the pros and cons of each method. Amongst the methods described in this chapter, we have focused on the mesh motion techniques based on the fixed connectivities, knowing their constraints and trying to lift them in order to improve their capabilities. One possible approach to avoid the mesh breaking down on fixed topology techniques in physical domain is allowing the cells slide over the boundaries using a slip condition model. This method has been explored in Chapter 3, but still needed some modifications to overcome the difficulties that are encountered in that chapter.
To do that, the problems of sliding the cells over the boundaries are solved by managing the mesh motion around a set of generic boundaries in the computational space following by mapping the computational mesh to the physical domain using a proper mapping operator.
Thereafter, various mapping operators are introduced and solved on unstructured meshes in Chapter 4

and the resulting meshes are compared with respect to their quality and smoothness. Chapter 5 studies a novel method to slide the mesh in computational space following by mapping the computational mesh to the physical domain at each evolving step.

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

CHAPTER 2

MESH MOTION IN PHYSICAL DOMAIN

In this chapter various mesh movement strategies in physical space are reviewed and studied. This is followed by specific examples in order to explore their characteristics and to demonstrate their applications. Some of these strategies are designed to produce a valid mesh given a set of prescribed boundary displacements by recomputing new mesh point coordinates without altering the connectivity or topology of the mesh which adds considerable capability for moving boundary problems. These methods can be divided into PDE based and algebraic techniques.

2.1 Mesh deformation based on fixed connectivity

In all mesh deformation techniques used in transient problems, the computational mesh is adjusted to the physical boundaries and is updated at every time step. Motion of all interior nodes in the physical domain is based either on the prescribed boundary motion, or the boundary and following that, the mesh motion is carried out by solving the problem.

2.1.1 Laplace equation

Generally, the proposed strategy for adaptive modification for time evolving meshes is divided up into four parts:

- move the domain boundaries
- apply the boundaries velocity to the internal grid nodes
- update the nodal positions
- grid management of special nodes

Given the coordinates of node I at time t as X_I^t, then the new coordinates X_I^{t+1} at time $t+1$ are obtained by the addition of displacement d_I^{t+1},

$$X_I^{t+1} = X_I^t + d_I^{t+1} \tag{2.1}$$

The method of computing the mesh displacement or deformation d_I^{t+1} is based on the solution of a boundary value problem using the following Laplace equation,

$$\nabla^2 \vec{P} = 0 \tag{2.2}$$

As part of a preliminary exploration, two ways to evaluate d_I^{t+1} were investigated; one is based on solving Eqn. 2.2 where $\vec{P} = \vec{X}$ represents the nodal position, and another using $\vec{P} = \vec{\omega}$, the velocity field, as proposed by Illinca et al. (1995).

As it is known, the nature of the Laplace equation is that the point displacements will be largest close to the moving boundary and diminishing for large distances. In addition, the Laplace equation locates each node as the average position of its surrounding points. Basically, the results obtained by the coordinate-based ($\vec{P} = \vec{X}$) Laplace equation gives the new position of the grid, and it is similar to the spring analogy where all edges are considered as a network of springs and the final solution is achieved when the springs reach their equilibrium state.
In the second method, the updated position for each node is obtained using the solution of the velocity field and a specified time step, Δt,

$$X_I^{t+1} = X_I^t + \Delta t \vec{\omega}_I^t \tag{2.3}$$

The main advantage of this method is the preservation of the mesh topology at each time step. This technique works well in convex regions, but it may produce poorly-shaped or even inverted elements in concave regions. Typically, the mesh surrounding concave geometrical items is pulled outwards leading to geometry-overlaying meshes. Due to excessive distortion, such meshes are usually quite difficult to recover via a posteriori quality improvement tools. Therefore, application of this approach

in its original form to meshes generated for arbitrary domains is generally not entirely adequate. Additional controls are necessary to prevent elements from being distorted during smoothing [Kovalev (2005)].

Fig. 2.1 shows the mesh at maximum displacement (before inversion) resulting from the linear motion of a body (circle), using both the solution of Eqn. 2.2 cast as a displacement (Fig. 2.1(c)) and the solution of the same equation cast as velocity (Fig. 2.1(d)). The comparison of the resulting meshes in Fig. 2.1(b) shows that using the velocity to update the new position of interior nodes, extends the applicability of the procedure to a displacement approximately 40% higher than the circle radius.

The main reason why the method is stiff and unable to handle large motion is that both smoothing methods yield the same direction field for the displacement vectors. Fig. 2.2 illustrates how the interior nodes do not have the freedom to adapt their position according to the boundary's curvature. This velocity field has been determined by setting $\vec{V} = (1, 1)$ on the inner boundary nodes as a boundary condition, which yields a stiff condition for displacement. Hence, the mesh loses its validity for large amplitude of motions.

2.1.2 Linear elasticity

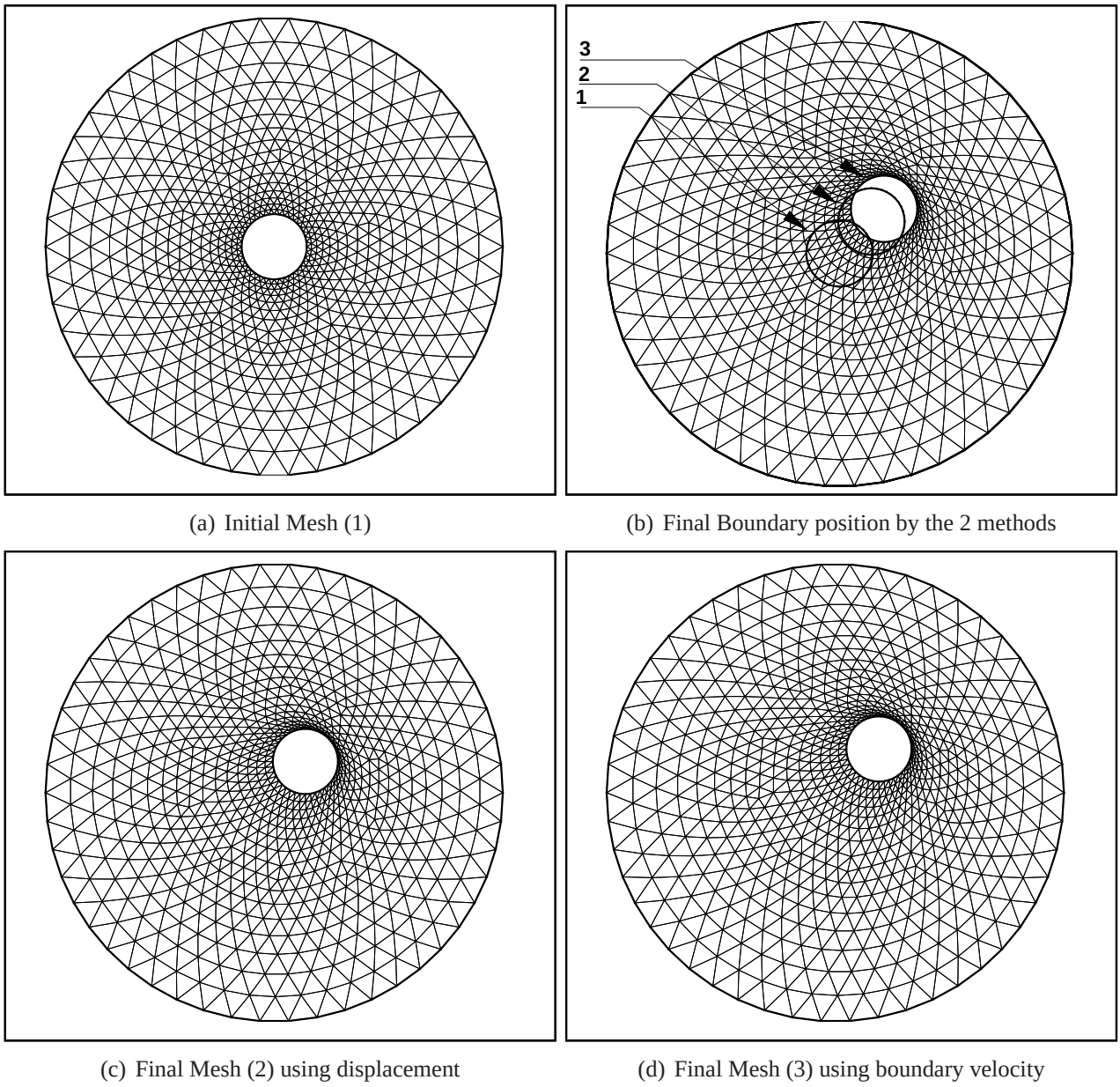
A recent formulation, presented by Stein et al. (2003), for moving boundary and mesh deformation problems can be obtained using a model based on an analogy for the linear elastic behavior of a material where the strain is proportional to the stress on the element.

The mesh is computed from the linear elasticity equations, which can be written as:

$$\begin{aligned} \nabla^2 u + \frac{1}{1-2\nu}\frac{\partial}{\partial x}\nabla \cdot V &= 0 \\ \nabla^2 v + \frac{1}{1-2\nu}\frac{\partial}{\partial y}\nabla \cdot V &= 0 \end{aligned} \tag{2.4}$$

where the nodal displacement vector is given by $V = u\hat{i} + v\hat{j}$. The parameter ν (Poisson's Ratio[1]) in the denominator is typically set so that the coefficient $1/(1-2\nu)$ is equal to the aspect ratio of the local

[1]Poisson's Ratio is defined as the ratio of lateral strain to axial strain and is always positive in sign.



(a) Initial Mesh (1)

(b) Final Boundary position by the 2 methods

(c) Final Mesh (2) using displacement

(d) Final Mesh (3) using boundary velocity

Figure 2.1 Comparison of mesh displacement using boundary displacement and velocity, computed using Laplace equation

cell. This produces stiffness in regions with high aspect ratio cells and ensures that boundary-layer elements track closely the local boundary as it moves. The solution to these equations is a vector field defining the displacement of each node. Yang & Mavriplis (2005) showed how linear-elastic smoothing can be used to perform very large deformations of inviscid and viscous meshes. Their implementation uses a different form of the linear-elastic relations shown in Eqn. 2.5, for which the modulus of elasticity is allowed to vary and Poisson's ratio remains constant. One advantage of this

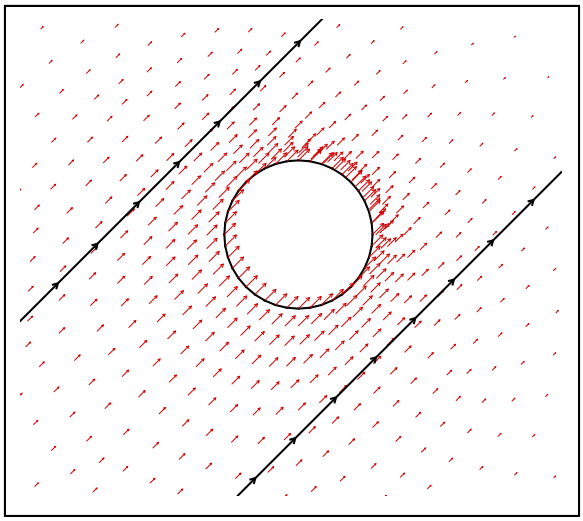


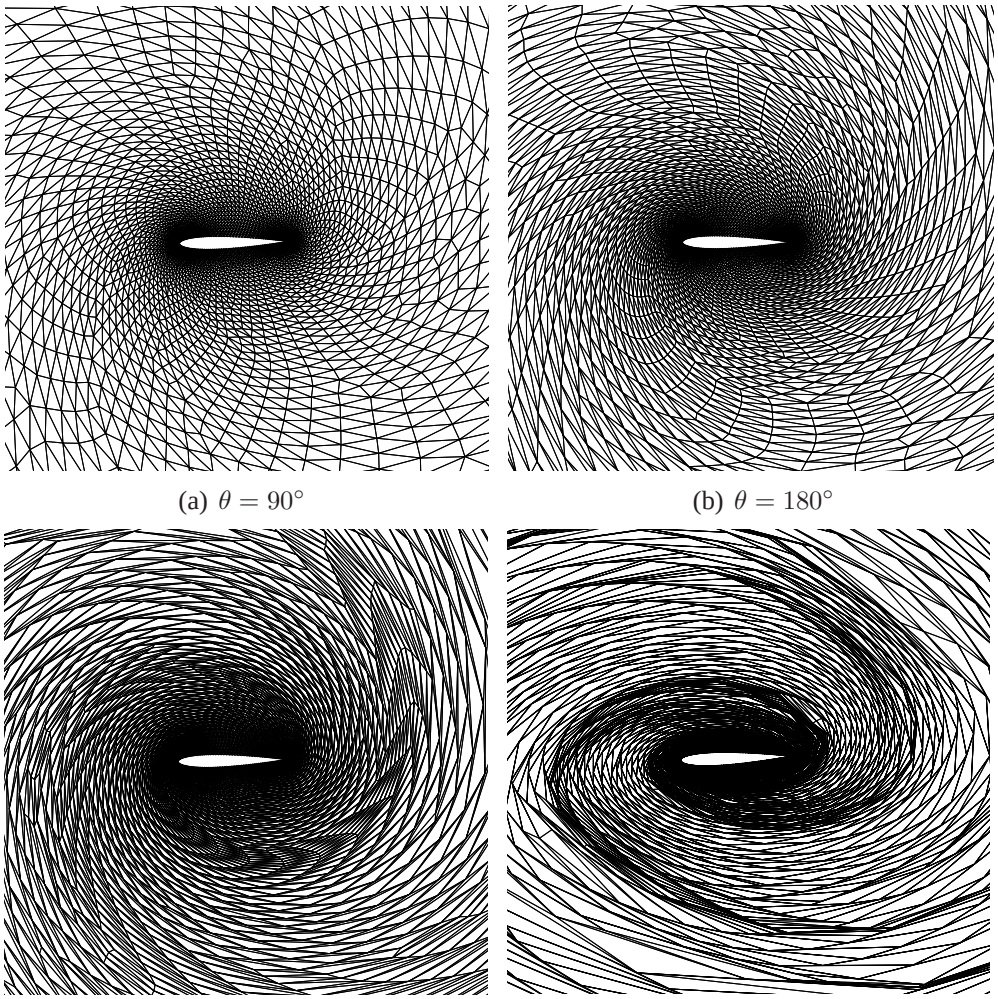
Figure 2.2 Propagated mesh velocity vector on the interior nodes

approach, is that in regions of large E (Young's modulus or modulus of elasticity) the mesh cells are displaced as a solid body. Thus, an appropriate prescription of the distribution of E can be used to avoid severe mesh deformation in critical regions of the mesh. With a distribution of E inversely proportional to the cell volume or to the distance from the deforming boundaries, much of the mesh deformation can be relegated to regions where the mesh is coarser and can sustain larger relative deformations.

$$\begin{aligned} \frac{\partial}{\partial x}[\frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\frac{\partial u}{\partial x}]+\frac{\partial}{\partial y}[\frac{E}{2(1+\nu)}\frac{\partial u}{\partial y}]+ & \\ \frac{\partial}{\partial x}[\frac{E\nu}{(1+\nu)(1-2\nu)}\frac{\partial v}{\partial y}]+\frac{\partial}{\partial y}[\frac{E}{2(1+\nu)}\frac{\partial u}{\partial y}] &= 0 \\ \frac{\partial}{\partial x}[\frac{E}{2(1+\nu)}\frac{\partial v}{\partial x}]+\frac{\partial}{\partial y}[\frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\frac{\partial v}{\partial y}]+ & \\ \frac{\partial}{\partial x}[\frac{E}{2(1+\nu)}\frac{\partial u}{\partial y}]++\frac{\partial}{\partial y}[\frac{E\nu}{(1+\nu)(1-2\nu)}\frac{\partial u}{\partial x}] &= 0 \end{aligned} \tag{2.5}$$

The moving procedure is applied by finding the value of V on the cell nodes and updating their position using this computed velocity field. This method is particularly advantageous when several boundaries move in different directions. Fig. 2.3 shows a NACA0012 airfoil rotated by 360° in four equal steps of 90°. It has been shown by this figure, that the method allow extremely large

deformations of poor quality meshes.



(a) $\theta = 90°$ (b) $\theta = 180°$

(c) $\theta = 270°$ (d) $\theta = 360°$

Figure 2.3 Incremental rotation of a NACA0012 airfoil in $360°$ using the linear elasticity model [figure courtesy of professor R.P. Dwight TU Delft, Dwight (2006)]

2.1.3 Radial basis functions

Another method that has been studied in the current work is Radial Basis Functions interpolation (RBF). In moving boundary problems, the displacement of all interior points in the physical domain are interpolated from the boundary positions at each time step. In the RBF formulation, the interpolation function $s(X)$ which describes the displacement of the interior nodes is

$$p(X) = \sum_{i=1}^{i=N_b} \alpha_i \phi(\| X - X_i \|) \tag{2.6}$$

where $p(X)$ is the function to be evaluated at X, ϕ is a given basis function similar to a function of distance field. The nodes X_i represent the known boundary value displacements, and N_b is the number of boundary points. In this method, node i identifies the center for a RBF. While X_i is the location of that center, the coefficient α_i is found at each point by solving a global interpolation problem. The RBF interpolating function, Eqn. 2.6, can be conveniently expressed as matrices when considering the entire set of mesh points. Using "s" to denote a boundary mesh point.

$$x_s = C_{ss} a_x \tag{2.7}$$

$$y_s = C_{ss} a_y \tag{2.8}$$

where

$$x_s = \begin{pmatrix} x_{s_1} \\ \vdots \\ x_{s_{N_b}} \end{pmatrix}, \quad y_s = \begin{pmatrix} y_{s_1} \\ \vdots \\ y_{s_{N_b}} \end{pmatrix}, \quad a_x = \begin{pmatrix} \alpha^x_{s1} \\ \vdots \\ \alpha^x_{s_{N_b}} \end{pmatrix} \tag{2.9}$$

The C_{ss} is expressed by

$$C_{ss} = \begin{pmatrix} \phi_{s_1 s_1} & \phi_{s_1 s_2} & \cdots & \phi_{s_1 s_{N_b}} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{s_{N_b} s_1} & \phi_{s_{N_b} s_2} & \cdots & \phi_{s_{N_b} s_{N_b}} \end{pmatrix} \tag{2.10}$$

which is a $N_b \times N_b$ matrix and its components are defined as

$$\phi_{s_i s_j} = \phi(\| X_{s_i} - X_{s_j} \|) \tag{2.11}$$

Some basis functions are presented in Table 2.1, where $\phi(\| x \|) = 0 \; \forall \; \| x \| \geq 1$, and more can be found in Rendall & Allen (2008). Choosing the proper basis function provides the combination of mesh quality and matrix conditioning.

To determine the dependence of N interior points on the N_b boundary points, the following matrix must be formed, where v indicates an interior node

Table 2.1 Basis functions for $\| x \| < 1$

Name	Definition $\phi(\| x \|)$
Wendland's $C0$	$(1- \| x \|)^2$
Wendland's $C2$	$(1- \| x \|)^4(4 \| x \| +1)$
Wendland's $C4$	$(1- \| x \|)^6(35 \| x \|^2 +18 \| x \| +3)$
Wendland's $C6$	$(1- \| x \|)^8(32 \| x \|^3 +25 \| x \|^2 +8 \| x \| +1)$

$$A_{as} = \begin{pmatrix} \phi_{v_1 s_1} & \phi_{v_1 s_2} & \cdots & \phi_{v_1 s_{N_b}} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{v_N s_1} & \phi_{v_N s_2} & \cdots & \phi_{v_N s_{N_b}} \end{pmatrix} \tag{2.12}$$

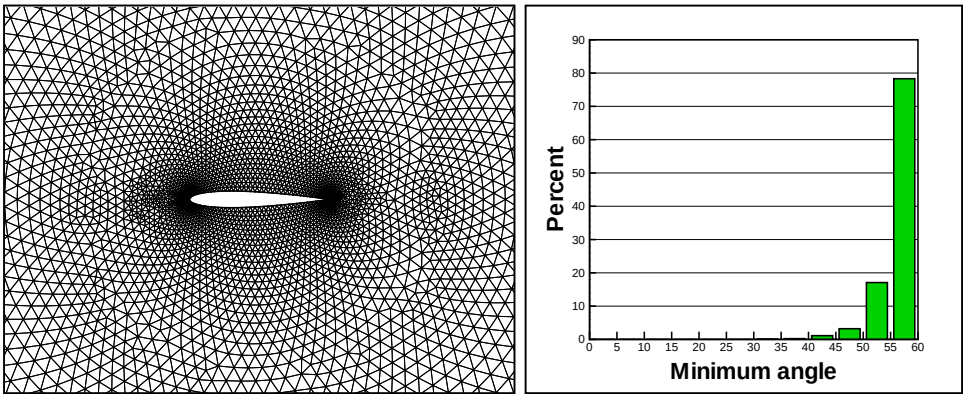
The positions of the domain interior points given by the vectors x_v and y_v are

$$x_v = A_{as} a_x = A_{as} C_{ss}^{-1} x_s \tag{2.13}$$

$$y_v = A_{as} a_y = A_{as} C_{ss}^{-1} y_s \tag{2.14}$$

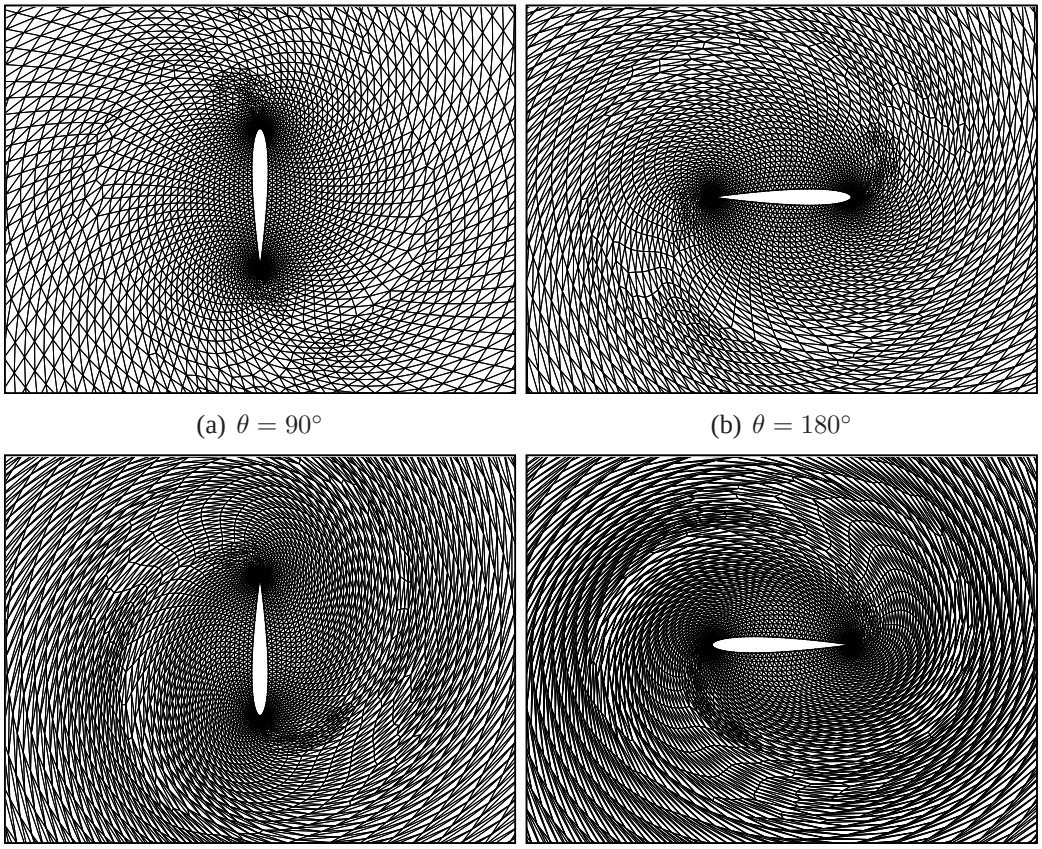
In the original RBF methods, the C_{ss} matrix is created based on the undisturbed boundary points from the initial mesh and the interior nodes are updated by that matrix at each step. In this work the C_{ss} matrix is created based on the previous displaced boundary to handle large amplitudes and motions. The influence of updating C_{ss} at each time step is studied in the following two examples.

This method was applied to rotate two different geometries, a NACA 0012 airfoil and a four-petal rose for exceptionally large deformations by taking several intermediate steps. The initial mesh for the NACA 0012 airfoil was generated for an incidence angle, $\theta = 0°$, shown in Fig. 2.4(a). The mesh quality defined as the distribution of minimum angle is shown in Fig. 2.4(b). The airfoil is then rotated about its centroid by a total of $360°$. Fig. 2.5 shows the resulting mesh at four angular positions. As the boundary cells remain attached to the body, the cells in the domain become highly skewed through this large deformation. Although the mesh maintains its validity, the quality will decrease as shown in Figs. 2.6.



(a) (b)

Figure 2.4 Initial mesh around NACA0012 and mesh quality

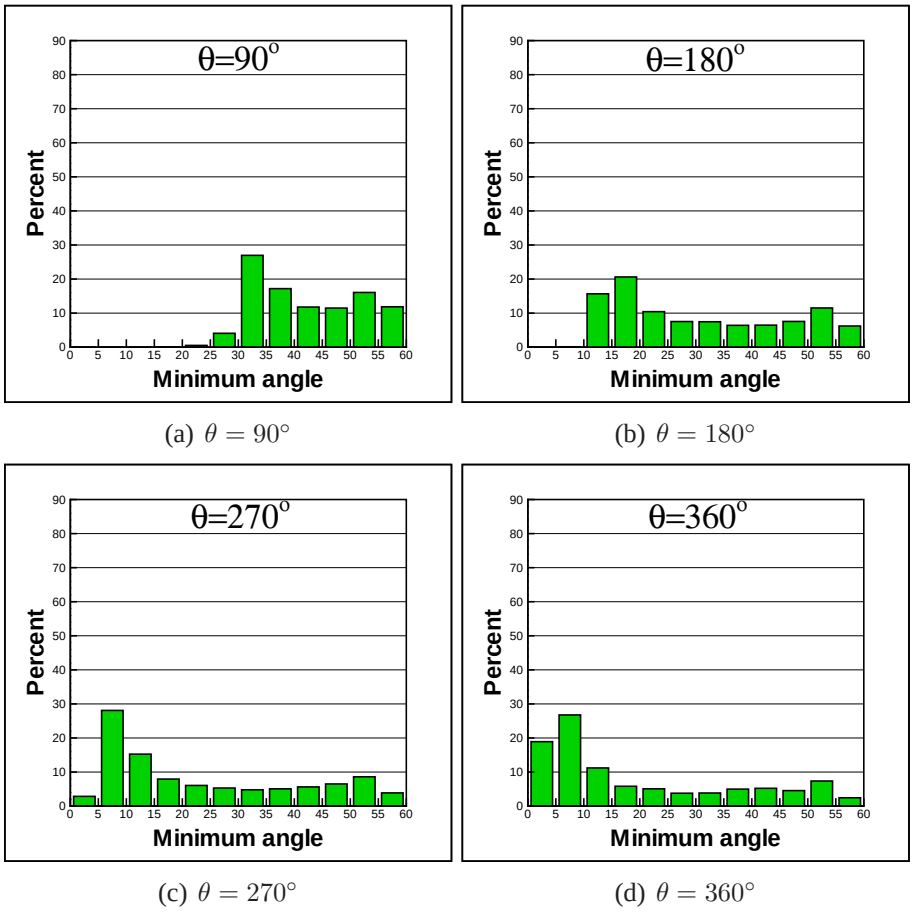


(a) $\theta = 90°$ (b) $\theta = 180°$

(c) $\theta = 270°$ (d) $\theta = 360°$

Figure 2.5 Deformed mesh around a rotated NACA0012 airfoil at four angular position using Radial Basis Functions interpolation method

In the second test case, the more complex geometry of a nonconvex boundary of a four-petal rose configuration is investigated. The initial mesh and its quality are given in Fig. 2.7(a) and (b). The



(a) $\theta = 90°$ (b) $\theta = 180°$

(c) $\theta = 270°$ (d) $\theta = 360°$

Figure 2.6 Mesh quality at four angular positions for the rotated NACA0012 airfoil

cross is rotated $360°$ in the clockwise direction around its centroid. Figs. 2.8(a)-(d) show the resulting mesh at four angle positions, and the corresponding mesh quality is given in Figs. 2.9. By comparing the undeformed and deformed meshes, it can be seen that the isotropy of the cells decreases with increasing rotation, and the cells gradually become elongated. This elongation yields very poor mesh quality at the final step.

2.2 Mesh movement by changing connectivity

At the end of the mesh deformation procedures described in the previous sections, some elements in the domain become flattened and stretched, so that the method can no longer produce a valid grid. Each pair of such elements can be cured by a swapping procedure which changes the skewed and flattened elements to more isotropic triangles and allows the motion process to continue. This

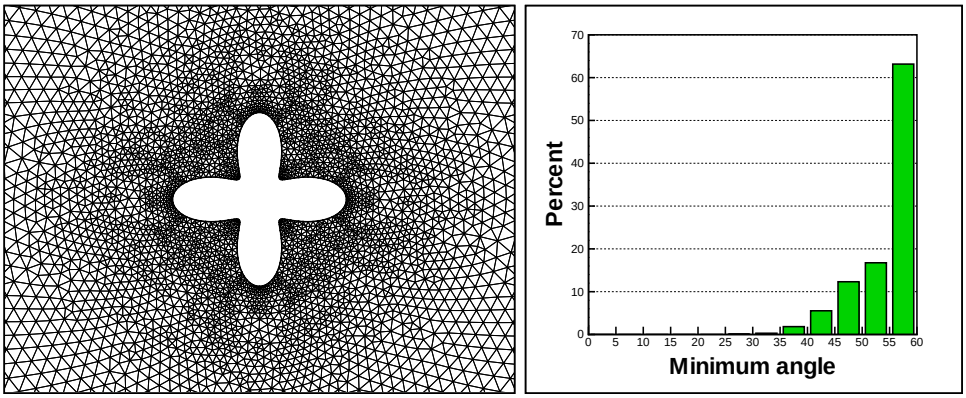
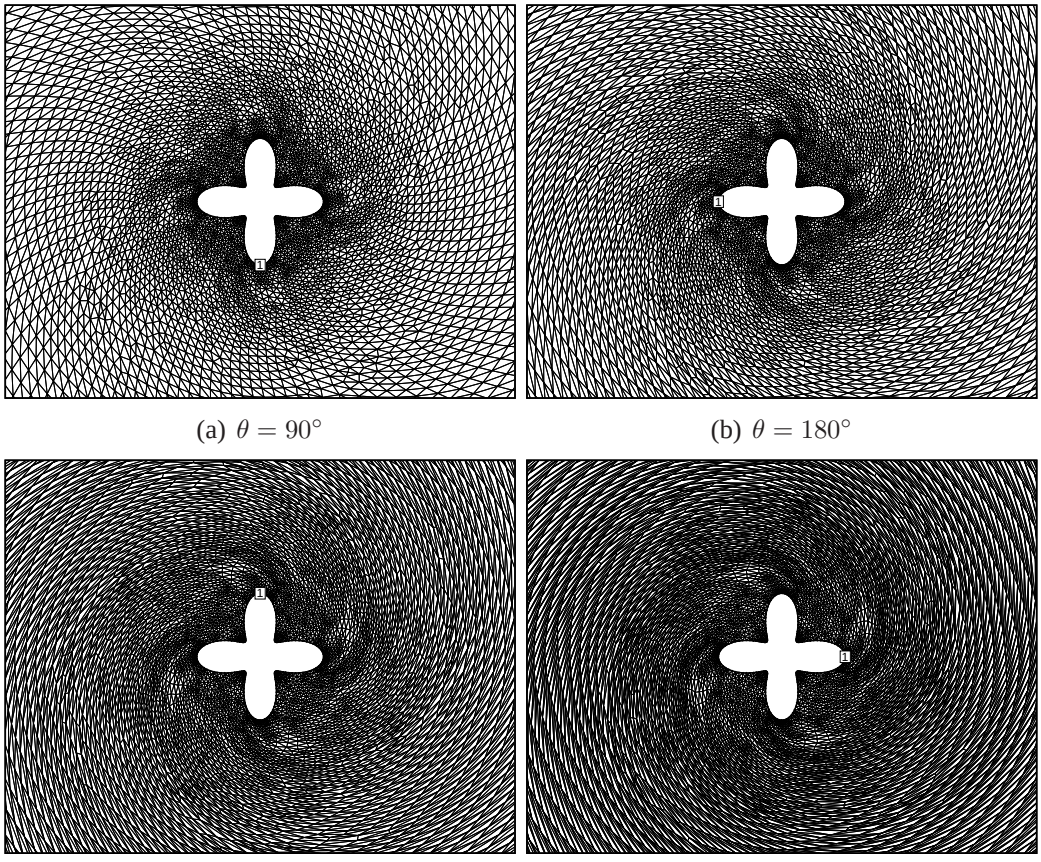


Figure 2.7 Initial mesh around a four-petal rose and mesh quality



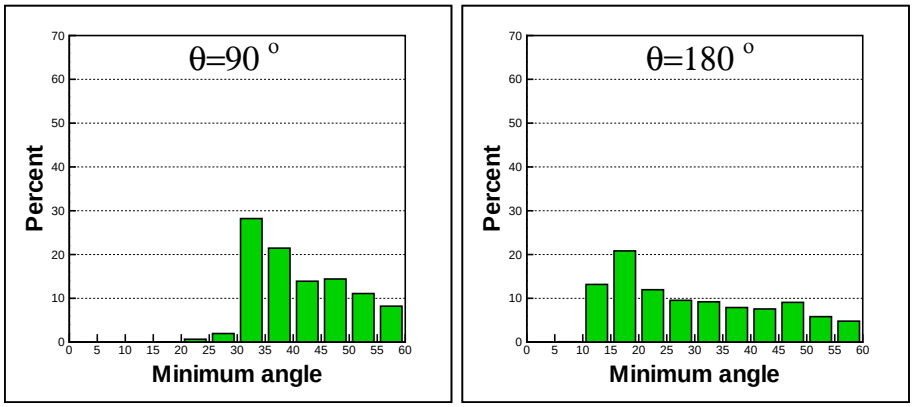
(a) $\theta = 90°$ (b) $\theta = 180°$

(c) $\theta = 270°$ (d) $\theta = 360°$

Figure 2.8 Deformed mesh around a rotated four-petal rose at four angular position using Radial Basis Functions interpolation method

requires the establishment of a criterion based on grid quality for the application of the curing step.

Fig. 2.10 shows how the application of this swapping procedure can improve and cure a highly



(a) $\theta = 90°$ (b) $\theta = 180°$

(c) $\theta = 270°$ (d) $\theta = 360°$

Figure 2.9 Mesh quality at four angular positions for the rotated four-petal rose

stretched grid with elements piled up ahead of the moving boundary, as presented in Section 2.1.1. This procedure is then followed by a smoothing step based on Laplace's equation to improve mesh quality.

To asses this, the method was implemented to a moving and deforming body shown in Fig. 2.11. As it is depicted, the method can handle any large motions in time with changing topology and high curvatures as well.

This method has been developed and extended successfully to 3D by Alauzet & Olivier (2011) where the mesh is smoothed after the swapping procedure using an elasticity-like equation and a mesh optimization technique.

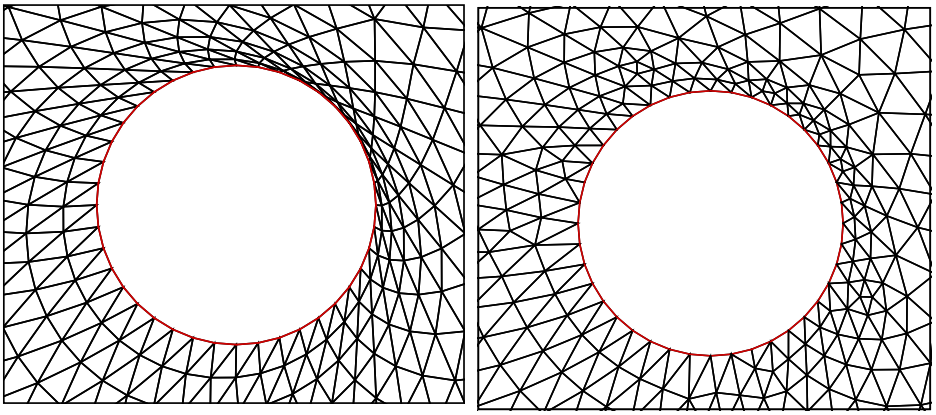


Figure 2.10 Swapping the flattened mesh to avoid mesh breakdown

2.3 Discussion

In all methods presented in this chapter, the management of the mesh movement takes place in physical space either through fixed or changing topology.

Amongst the mesh deformation techniques for mesh motion with fixed topology, Laplacian type smoothing restricts nodal displacement to a certain limit in order to avoid sever element distortion and, as such, it is not suitable for the objectives of this work, which is to handle arbitrary large mesh motion. Poisson equations can extend the amplitude of the motion by an appropriate choice of the forcing terms. But, this approach deals with a posteriori information in forcing terms which decreases the prospect for automation of the method. Ideally, the forcing terms should be chosen so that all mesh points move in the domain according to the boundary shape and direction of motion. However, choosing these forcing terms needs several trials to achieve the best resulting mesh in physical space.

The linear elasticity approach is one of the most robust techniques among PDE based methods for large mesh motion problems. Using this method maintains the mesh validity around concave corners in the physical domain as well.

The radial basis functions interpolation method is by far the most efficient method for large deformations and gives better resulting mesh quality than linear elasticity.

The strong point of all deformation techniques is the preservation of the connectivity of the mesh throughout the mesh motion, thus avoiding the use of interpolation techniques at each time step. However, all methods will eventually fail, because of the fact that the cells remain attached to the boundaries during the motion.

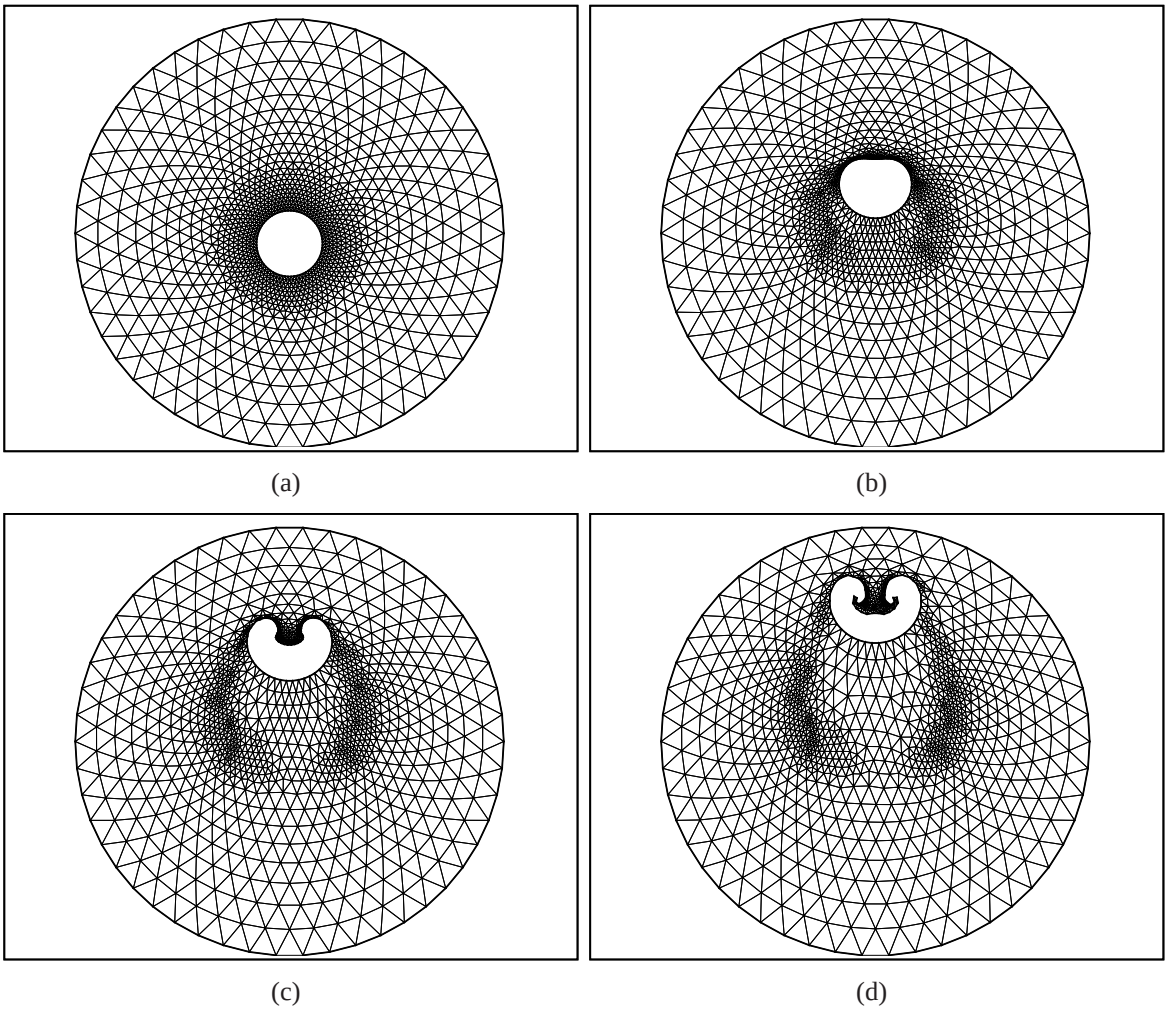


Figure 2.11 Moving and deforming boundary in domain based on swapping edges and smoothing nodes

One solution to this problem is by swapping edges to cure the flattened cells. This maintains the number of nodes, edges and cells, and can be applied for complex motions in both 2 and 3D unsteady problems. However, this does not avoid interpolation altogether. Indeed, depending on the discretization techniques used by the flow solver, interpolation may still be required even after applying the edge swapping. This depend on how the primitive variables are stored, i.e. at cell centers or mid point of the edges.

An original solution to overcome this drawback, proposed in this work, is to allow the cells to slide past the boundaries. Removing the no-slip condition gives the mesh motion considerably increased degrees of freedom to realize the amplitude of the motion.

CHAPTER 3

MESH MOVEMENT USING SLIDING CELLS

This chapter presents a new method to generate moving grids with large amplitude for rigid body motions. This is based on an analogy of particles (grid cells) advected using a slip condition, where the cells slide over the boundaries. The context is one of large motion where the grid connectivity is preserved, except at specific nodes on the boundaries. In addition to having a suitable form for using ALE methods, the proposed algorithm is capable of handling approaching as well as separating boundaries. This boundary detection is achieved by defining a quasi-distance field in the domain to control the movement of interior nodes. Unlike other methods, the proposed approach leads to higher efficiency and precision by avoiding the need for interpolation of variables as time evolves, and it simplifies grid management in the physical domain.

3.1 A flow analogy for mesh motion

All the mesh deformation techniques investigated in Chapter 2 (except overset method) will finally fail for large boundary motions essentially because of the constraint whereby cells remain attached to the boundaries. The proposed solution to lift this restriction is to allow cells to slide over the boundaries using a slip condition. Moving the grids in this approach will be based on a model where each 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 schematically shown in Fig. 3.1. As a result, cells near the boundary will be deformed due both to the sliding motion and movement of the boundary. The global grid motion can now be reformulated as an extension of previous methods, using PDE models, such as the Laplace equation or linear elasticity model described in Chapter 2, with the extension of using a slip boundary condition. The modified method for adaptive modification for time evolving meshes will be based on the solution of the following B.V.P., using Laplace's equation

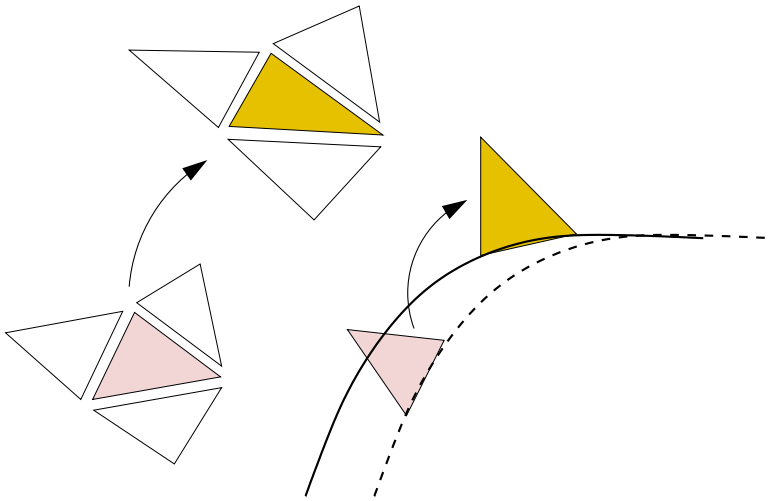


Figure 3.1 Sliding cells on a moving solid boundary

$$\nabla^2 \vec{P} = 0 \tag{3.1}$$

where P is the nodal position for interior nodes. This equation is solved, subject to the nodal positions of the boundary nodes, as shown in Fig. 3.1.

Figure 3.2 illustrates the direction of moving nodes around a boundary. As can be seen in this figure, in contrast to the directions shown in Fig. 2.2, the path lines for moving nodes near the body conform to the curvature of the boundary. These curvilinear directions are obtained by applying the slip condition and allowing the cells to slide over the boundary.

Two particular points shown in Fig. 3.2 require special treatment. These are the separation and the reattachment at the "leading" and "trailing" edges[1] where, two adjacent cells split or re-attach at the leading and trailing edges, respectively. After re-attachment, the body cells at the trailing edge are shed into the domain.
The procedure of this new approach consists in the following steps

1. Separating two contiguous elements along a common edge at the stagnation point

[1]The use of the terms leading and trailing edges in the present context are not strictly the same as the definitions in fluid mechanics.

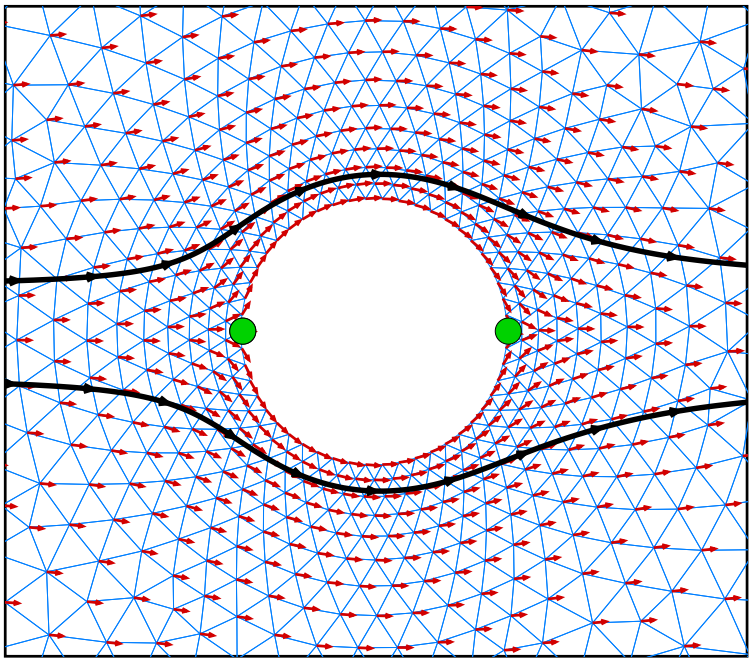


Figure 3.2 Generated mesh and direction of motion of the cells past on a circle

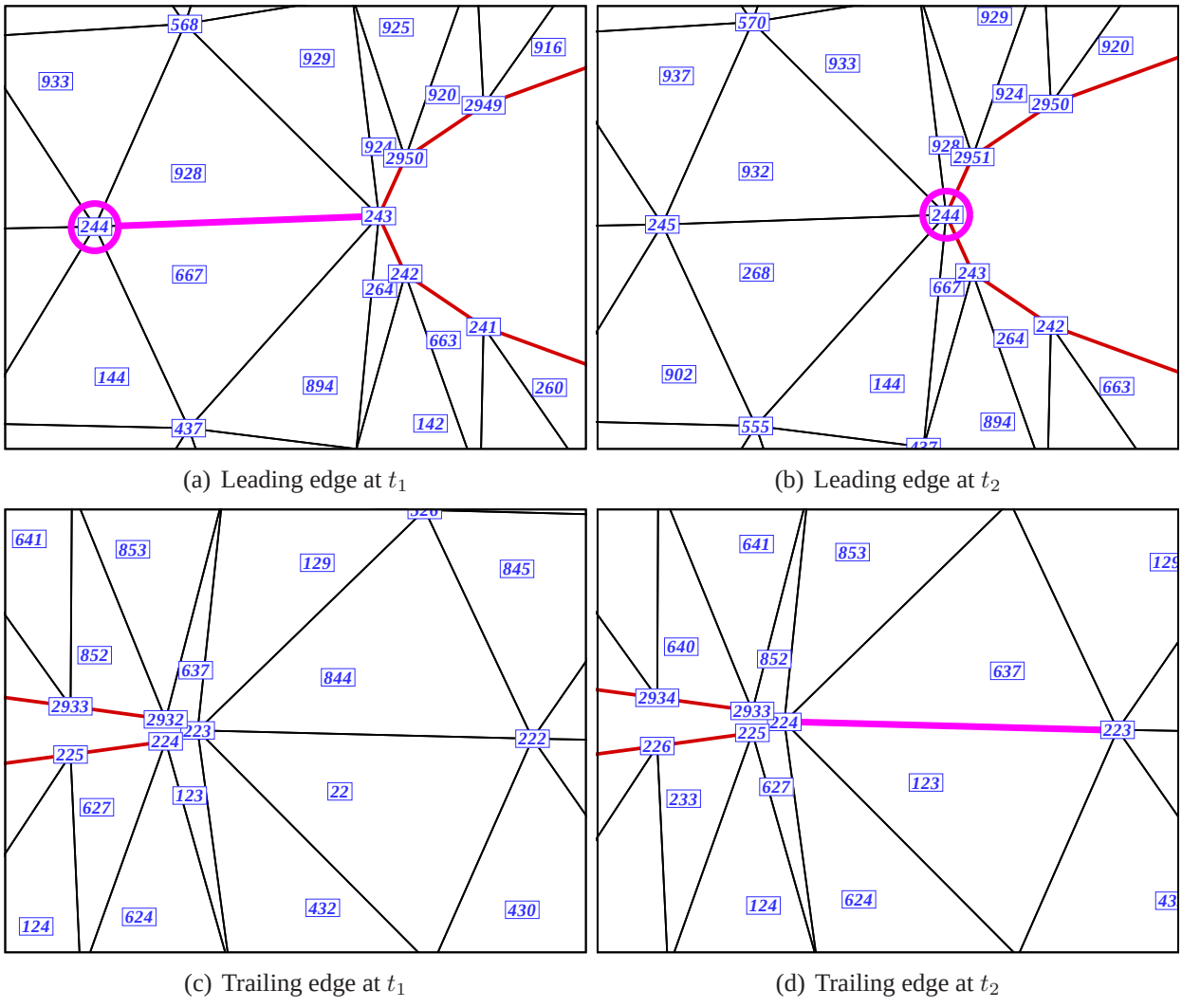
2. Sliding the nodes on the boundary

3. Re-attaching the elements which are leaving the trailing-edge

4. Smoothing the mesh.

Figures 3.3 shows the sliding procedure at the leading and trailing edges as the boundary moves in the $(-x, 0)$ direction. In Fig. 3.3(a) the candidate node and edge are marked according to the direction of the motion. The candidate node ahead of the boundary will replace the node of the leading edge and two neighboring cells will be detached from the candidate edge shared between them. The next step is to add a node with the same coordinate as the leading edge node, 243, to the nodes list which, for this specific example, is node 2951. In the subsequent displacement steps, both nodes 243 and 2951 will slide along their respective boundaries at the same pace. Fig. 3.3(b) shows how node 244 that attaches to the leading edge and two adjacent cells, 928 and 667, are split. The rest of the nodes on the upper and lower surfaces are moved along the boundary, and toward the trailing edge.

At the trailing edge, the candidate node is shed into the domain, and the two elements are combined along the two boundary edges. Fig. 3.3(c) depicts node 233 leaving the trailing edge, and nodes 224 and 2932 which are merged at the trailing edge, as shown in Fig. 3.3(d). After all these modifications,



(a) Leading edge at t_1

(b) Leading edge at t_2

(c) Trailing edge at t_1

(d) Trailing edge at t_2

Figure 3.3 Treatment of the leading and trailing edges

a Laplace equation-based smoothing on the new position of the boundary is solved to yield the new nodal positions of the entire mesh. During this procedure, the number of nodes, cells and the edges remain constant and the method preserves the connectivity matrix, except around the leading and trailing edges.

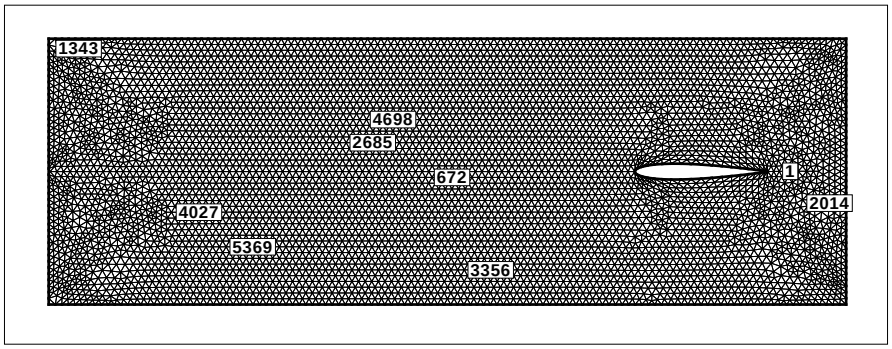
3.2 Application of the method

The method has been applied for an airfoil sliding horizontally in the $(-x, 0)$ direction and the result is shown in Figs. 3.4 at four different steps. In this figure, one can follow tagged sample points as the airfoil passes through the mesh. In particular, one can observe that the mesh resumes its previous geometry and topology after the passage of the body.

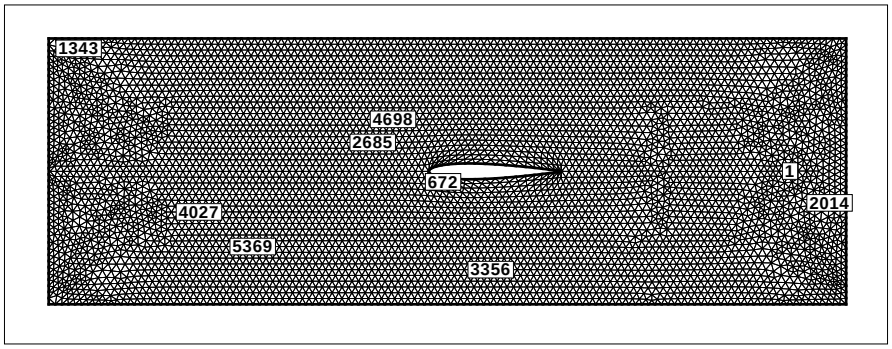
This was applied extensively to test the procedure and found to work adequately for slender bodies. However, in the case of a bluff body moving through the domain, this approach results in a stretching of the cells ahead of the body after just a few time steps. For instance, in the case of a circle moving in the $(-x, 0)$ direction shown in Fig. 3.5, stretched cells appear clearly after only three time steps.

This is essentially the same problem that was encountered with the no-slip boundary presented in Section 2.1.1. Namely that the moving body pushes the cells ahead of it, and these become flattened and eventually yield invalid meshes. At first sight, this is in contradiction with the objective of sliding meshes, as this approach was proposed as a way to avoid this problem. What is happening is that there are two effects at work: 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 away fast enough. So, the cure would be a restriction on the time step related to the cell size ahead of the stagnation point.

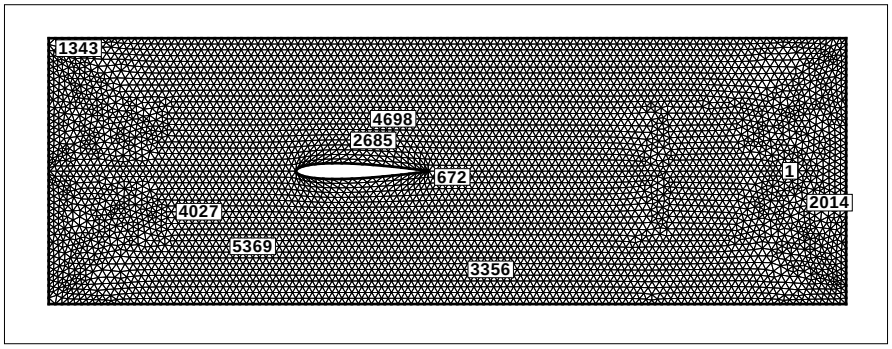
Another method to resolve this situation is to modify the model equation, so that the propagation of the boundary velocity is increased. Hence a proper forcing term to propagate the boundary's displacement to the interior nodes is required. The magnitude of this term can be based on a *distance parameter* which behaves like a distance field solution and can be carried out by solving Eq. 3.2. This



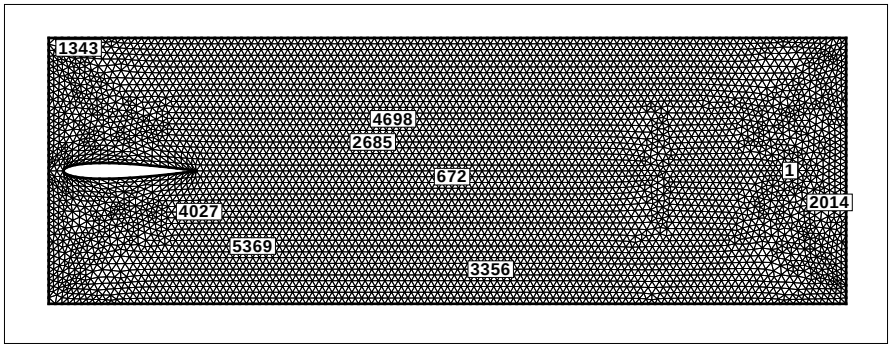
(a)



(b)



(c)



(d)

Figure 3.4 Sliding an airfoil through the cells

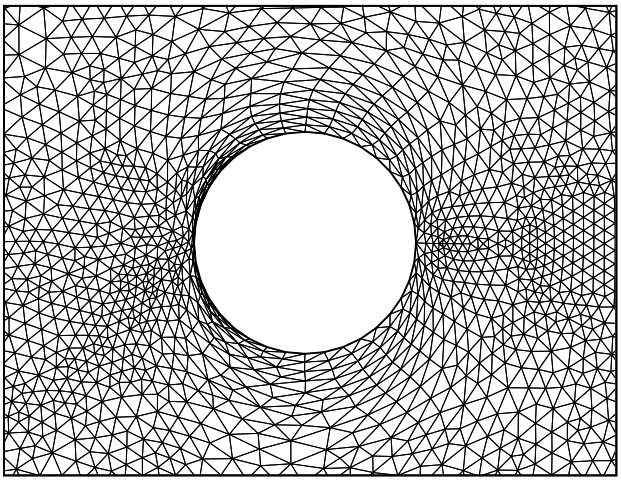


Figure 3.5 Created flattened cells ahead of a moving circle in $-X$ direction

is a much less computationally intensive task than computing the actual distance field.

$$\nabla(k\nabla\phi) = 0 \tag{3.2}$$

where $k_i < 1$ can be a constant number for all cells. This method casts the boundary propagation problem as the solution of a boundary value problem, Eq. 3.2, in which the front (boundary) to be offset is the isovalue $\phi = 0$ of a so-called quasi-distance field, and is viewed as the inner boundary condition. Then the values of ϕ are computed exactly for each node on the far field boundary as the distance to the closest point on the body. The iso quasi-distance curves which are equivalent to the given offset quasi distance are the resulting offset and need to be extracted from the computational domain. Fig. 3.6 and Fig. 3.7 depict the quasi-distance field ϕ and $\nabla\phi/|\nabla\phi|$ for the profile illustrated in Fig. 2.4. As can be seen from Fig. 3.7, the calculated unit vectors based on $\nabla\phi$ are naturally perpendicular to the body surface and yield a proper criteria for the prescribed forcing term. These iso-ϕ values are not exact distances from the boundary, but behaves approximately like the exact values which can be obtained by solving the Level-Set or Eikonal equations.

Generally, the updated position for each node in the moving boundary problem is obtained using

$$X_i^{t+1} = X_i^t + \Delta X_i \tag{3.3}$$

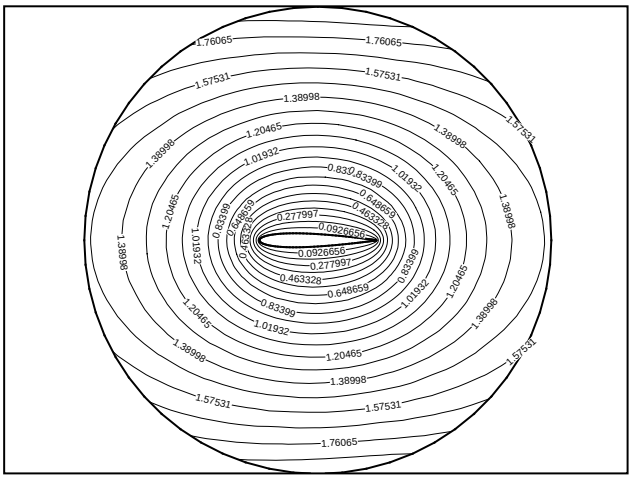


Figure 3.6 ϕ field around a NACA0012

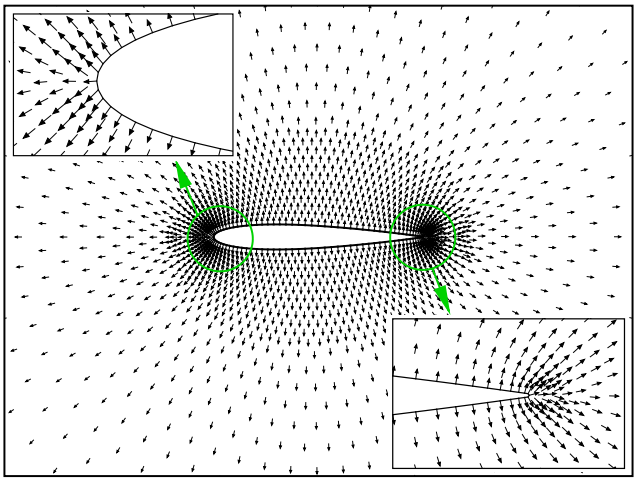


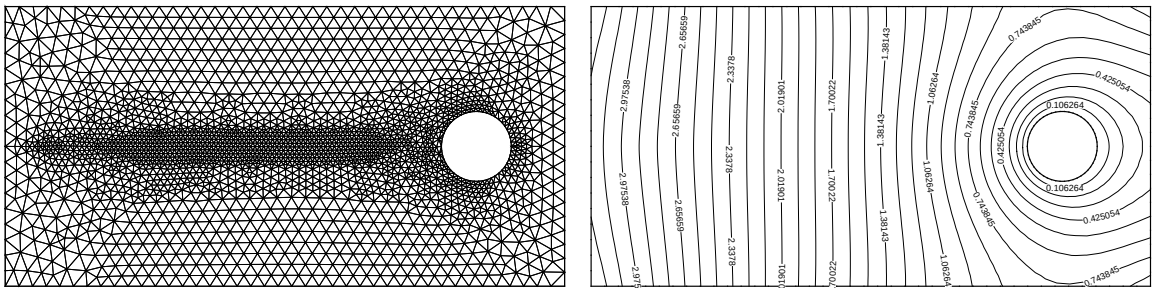
Figure 3.7 $\nabla\phi/|\nabla\phi|$ vectors around a NACA0012

In the present formulation, the update of the location of internal nodes was related to their distance from the boundary using

$$\Delta X_i = \alpha\vec{\omega}_b\phi\frac{\nabla\phi}{|\nabla\phi|} \tag{3.4}$$

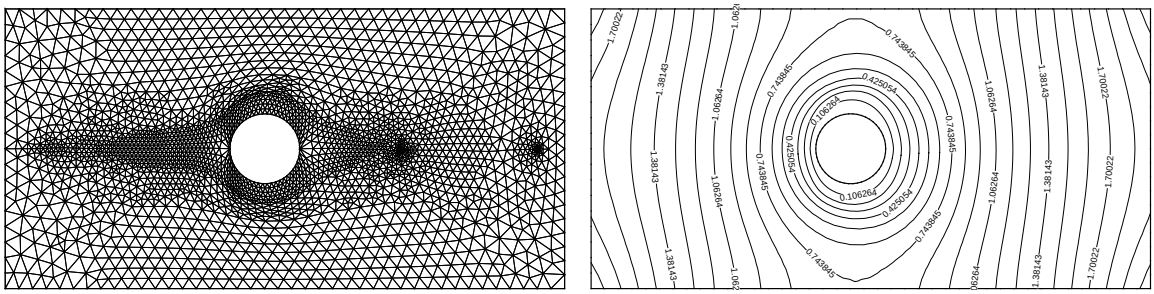
where α is a constant number and ω_b is the boundary's velocity. The entire procedure has been applied to the displacement of a cylinder in a cavity, while each node position updated using Eqn. 3.4 after sliding the cells on the boundary at each time step. Figs. 3.8, 3.9 and 3.10 present the results

when the cylinder is moving in negative x-direction at three different time steps, $t = 1, t = 50$ and $t = 100$ respectively. It is obvious that the time step has a fictitious definition in this problem and only represents the evolution procedure.



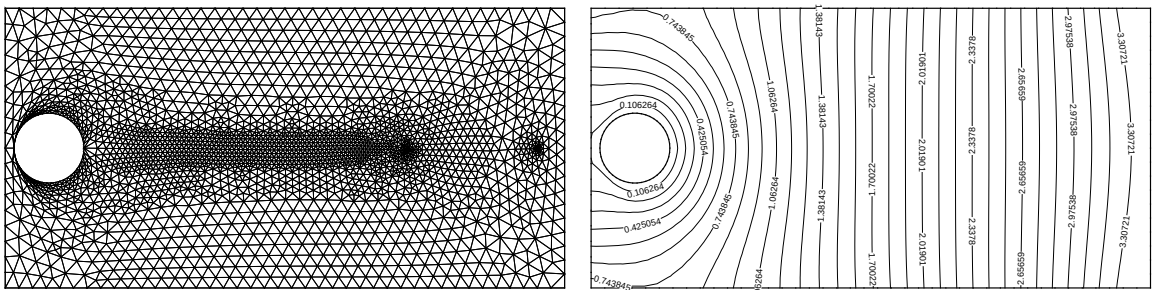
(a) (b)

Figure 3.8 The mesh (a) and ϕ (b) at $t = 1$



(a) (b)

Figure 3.9 The mesh (a) and ϕ (b) at $t = 50$

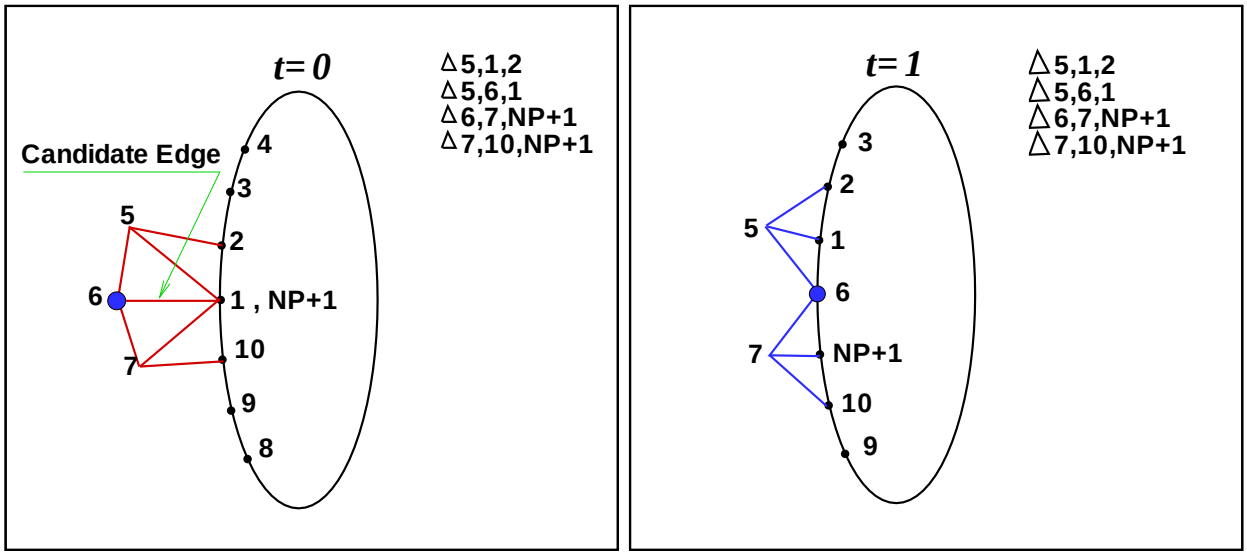


(a) (b)

Figure 3.10 The mesh (a) and ϕ (b) at $t = 100$

3.2.1 Restriction

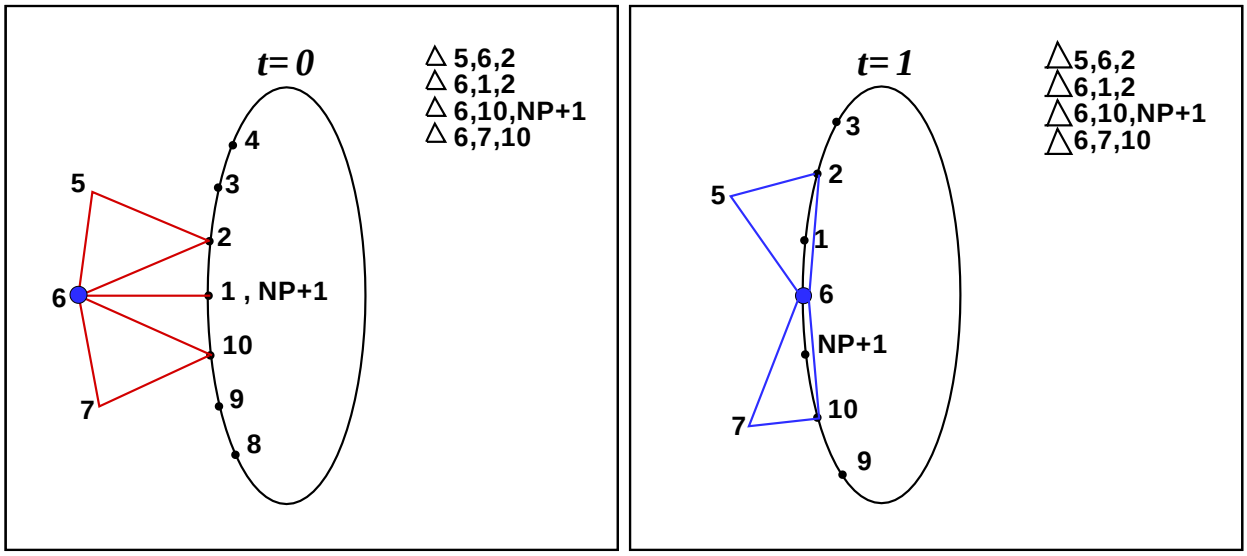
Two points in the mesh require special treatment; these are the separation and the reattachment at the "leading" and "trailing" edges. This is specially important for regions of the boundaries where the curvature is small i.e. bluff bodies. The process of separating two cells along their common edge may lead to invalid cells depending on the valence of the connectivity of the candidate node. The requirement is that, at the point of separation ("leading edge"), a pair of cells must have the proper valence (the number of edges and their arrangements attached to that point) which depends on the direction of the local motion of the boundary. The ideal configuration, shown in Fig. 3.11, is that three edges are attached to the leading edge (node 1). According to the direction of the motion, negative x-direction, node number 6 in Fig. 3.11 (a) is the candidate to be attached to the leading edge and cell(5,6,1) and cell(6,7,1) have to be separated by their common edge in the next time step, as shown in Fig. 3.11 (b). This kind of arrangement will result in a valid cell configuration after parting. In some cases,



(a) (b)

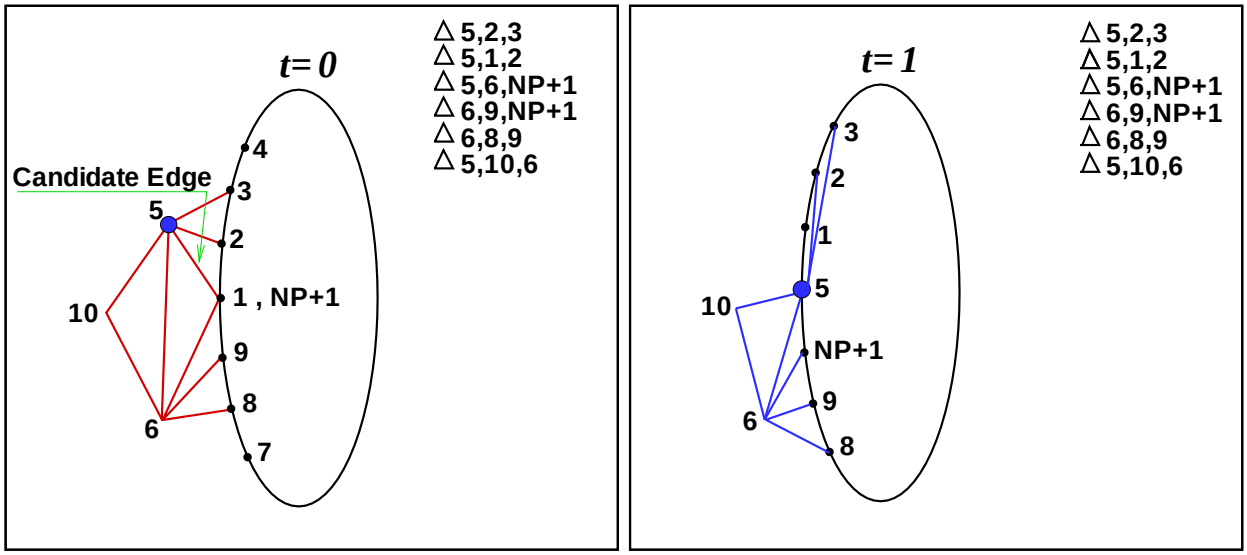
Figure 3.11 Ideal configuration for cell parting ahead of leading edge

separation of the cells ahead of the leading edge creates invalid cells on the boundaries. This usually happens when the leading edge is positioned on the part of the boundary where the curvature is a minimum. This situation is critical to the robustness of the scheme as rotation or deformation of the boundaries changes the position of the leading and trailing edges. Figs. 3.12 and 3.13 show two different pathological states which can be expected as time evolves. Again, in these figures it is



(a) (b)

Figure 3.12 Invalid cells after separation along a single edge



(a) (b)

Figure 3.13 Invalid cells after separation along a misaligned edge

assumed that the boundary is moving in the negative x-direction. Figs. 3.12(a) and Fig. 3.12(b) show how invalid cells may appear during the time evolution when only one edge is attached to the leading edge. In addition, Figs. 3.13(a) and 3.13(b) illustrate the situation when the slope of edges attached to the leading edge deviates substantially from the direction of motion.

To treat these undesirable states and reach a correct state, as illustrated in Figs. 3.11, swapping the edges and, in some cases, smoothing, has been applied locally. Fig. 3.14(a) depicts how local swapping can prepare the cells ahead of the leading edge at $t = 0$, so that at the next time step, shown in Fig. 3.14(b), an adequate configuration is encountered. At the trailing edge, where the cells re-attach

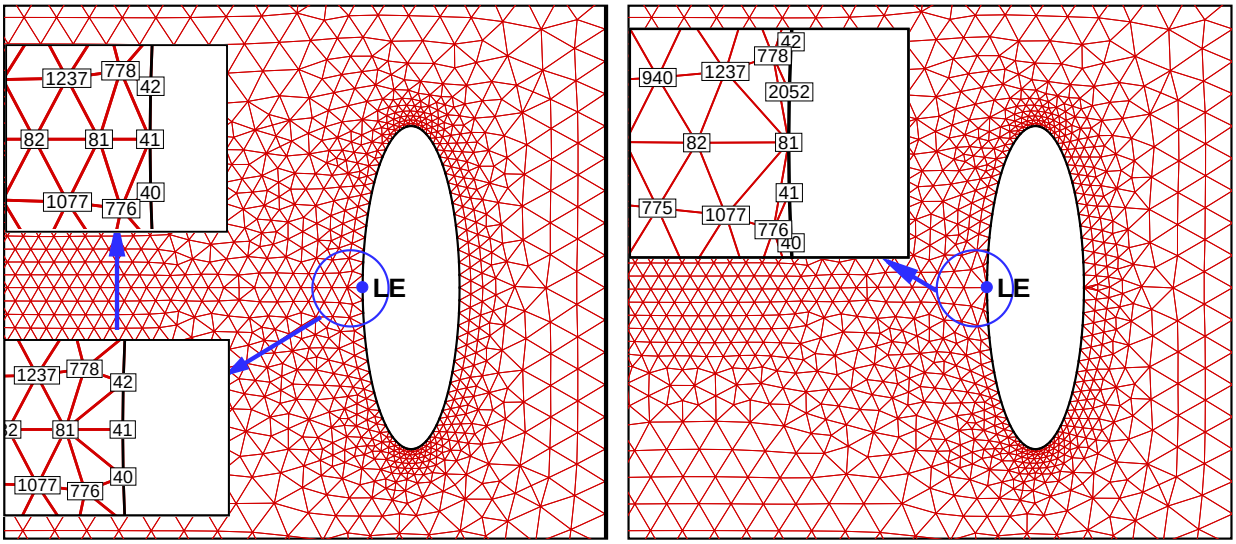


Figure 3.14 Swapping applied only on edge attached to the L.E.

like fluid particles leaving the boundaries in a potential flow field, an inverse procedure is applied.

3.3 Evaluation of the method

In this chapter the idea of sliding cells over the boundary has been proposed in order to release the major constraint in all mesh deformation techniques studied in Chapter 2. This constraint relates to the attachment of the cells on the boundary, as it is moving in the physical space.

While the sliding method presented in this chapter has proven to be a suitable tool for large motion applications, leading edge treatment in the bluff body motions and dealing with forcing terms de-

creases the robustness of the method.

Generally, the major drawbacks of the sliding mesh in the physical domain are the:

- propagation of body velocity in the domain and the definition of a proper time step or CFL-like number for the mesh,
- valence or arrangement of the edges attached to the leading edge node,
- complexity of computing the forcing term,
- weekness of Laplace equation to handle large mesh deformation.

To overcome these difficulties, it is proposed to manage the boundary motion and the grid movement in computational space rather than physical space. The grid motion can be transferred to the physical domain by applying a proper set of mapping operators at each moving step. This approach is developed in the next chapters and applied to complex configuration of bodies in relative motion.

CHAPTER 4

MAPPING DOMAINS

To alleviate some of the main issues identified in the application of the mesh sliding procedures when carried out in physical space, a new original approach is proposed, in which the mesh motion takes place in computational space, where grid management is simplified. The mesh in the physical domain is then obtained by a mapping of the computational mesh to the physical domain. Such mapping operators, which allow various types of grid control, have been used extensively in structured meshes, but these do not have direct extensions to unstructured grids.
In this chapter, the extension of various mapping operators to unstructured grids is studied, and several methods are compared with regards to mesh quality and smoothness.

4.1 *"A mesh to make a mesh"*

A widely used methodology to generate and to smooth meshes is to map an isotropic grid in 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 used in the mapping procedure in space $\mathcal{C}$, as illustrated in Fig. 4.1. Hence, a physical mesh is created through a transformation of a computational mesh.

Laplace's equation, in general form with various coefficients, is one of the most common mesh mapping schemes. Essentially, this consists in placing each node at the average position of its neighbors. However, around sharp corners in nonconvex regions, the method can yield inverted cells. Extensions, through Poisson equations with a set of control functions that serve as logical space weights are used to overcome this drawback. Their behavior, however, is difficult to predict and can still result in folded cells in extreme cases.

It is known that the Laplace equation is able to map any concave domain to a convex one. However,

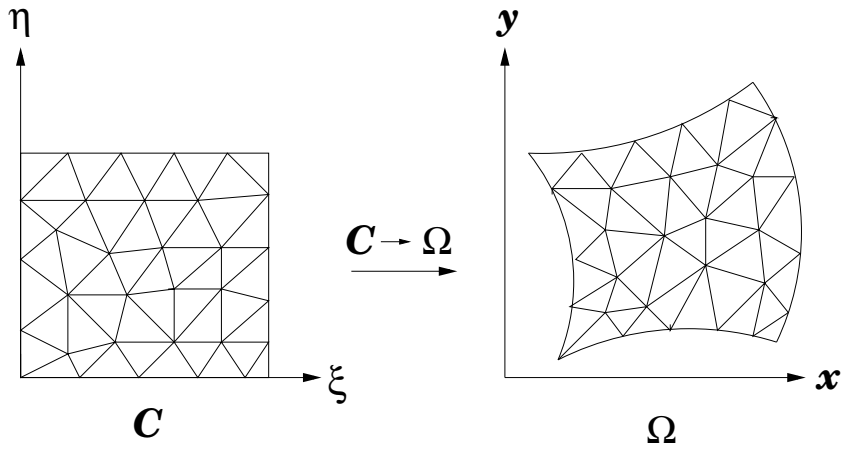


Figure 4.1 Mapping of a computational domain (ξ, η) to physical space (x, y) .

mapping a convex domain to a concave region using that equation gives a tangled mesh in most cases. This drawback can be avoided by reformulating this operator by inverting dependent and independent variables of the Laplace equation, giving a new form of the operator, named the Winslow equations. Solution to these equations gives a valid and smooth mesh in physical space with concave boundaries.

Spekreijse (1995, 1996) introduced a composite mapping which includes forcing terms to maintain orthogonality near boundaries. Similarly, Garon (1983), Khamayseh & Mastin (1996) and Khamayseh & Hansen (2000) have used Beltrami differentials on structured grids to control the area and angular distortion of the cells near physical boundaries. In Villamizar & Acosta (2009) and Villamizar & Weber (2007), a new system of elliptic operators have been derived to obtain uniform cell area by controlling the Jacobian along the curvilinear coordinate directions. However, these forcing terms are difficult to choose a priori to obtain a specific mapping behavior.

A more direct approach, where the desired characteristics of the resulting mesh can be established a priori is possible via variational methods. This approach for generating structured quadrilateral meshes for two dimensional domains has been proposed in Azarenok (2009), Castillo et al. (1988), Chibisov et al. (June 2006) and Khattri (2007). This approach is based on minimizing appropriate variational principles for controlling grid spacing, area and orthogonality.

4.2 Mapping Models

Generally, the mapping techniques based on PDEs, consist in solving these operators discretized by finite difference schemes on structured grids. In unstructured grids the conventional discretization by Green's integration schemes are not directly applicable. Because they are in nonconservative form, they do not have a natural extension to unstructured meshes. As a result, these methods were largely ignored for the generation and/or smoothing of unstructured grids in the physical domain.

In the present work, two mapping procedures are considered. In the first approach, the Winslow equations are solved on a triangulated computational space to transform the mesh from the computational to the physical domain. The second mapping procedure uses a system of PDE equations based on functionals, in order to get an improved transformation of the mesh in terms of its validity, quality in the physical domain, and conformity to the boundary shape. The Winslow equations are solved by two different discretization techniques 1) a modified finite difference scheme applicable on an unstructured grid, and 2) a finite volume formulation applied on the linearized mapping equations. For the functional based operators, only the finite volume technique has been applied.

4.2.1 The Winslow operator

Laplacian mapping, on both structured and unstructured meshes, can be achieved by the following system of partial differential equations in the computational domain $\mathcal{C}$,

$$\begin{aligned} x_{\xi\xi} + x_{\eta\eta} &= 0 \\ y_{\xi\xi} + y_{\eta\eta} &= 0 \end{aligned} \tag{4.1}$$

This system of equations yields valid meshes for regular geometries, but tangled meshes result for large variations of boundary curvature. The validity of this mapping can be extended by inverting the independent with dependent variables in Eqns. 4.1. Thus reformulated, this yields the Winslow (Winslow (1967)) equations in computational space (ξ, η),

$$\begin{aligned} \mathcal{L}(x) = \quad g_{11}x_{\xi\xi} - 2g_{12}x_{\xi\eta} + g_{22}x_{\eta\eta} &= 0 \\ \mathcal{L}(y) = \quad g_{11}y_{\xi\xi} - 2g_{12}y_{\xi\eta} + g_{22}y_{\eta\eta} &= 0 \end{aligned} \tag{4.2}$$

where

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

System of Eqns. 4.2 yields smooth meshes for most conventional convex shapes, but inappropriate mesh distributions can result for sharp concave corners where large changes of curvature occur. In Knupp & Steinberg (1993) and Knupp (1999) it is mentioned that while the Winslow operators guarantee continuum 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. Therefore, in order to devise a mesh mapping appropriate for arbitrary boundary shapes in the physical domain, in addition to positive Jacobians for all cells, other measurable criteria must also be considered.

4.2.2 Mesh mapping by functionals

In mapping methods based on functionals (Castillo et al. (1988)), a monitor function is defined to control the cells desired properties, specially near high curvature parts of the domain boundary. These operators are based on variational smoothing methods, and are derived by the minimization of appropriate grid functionals for area, length and orthogonality, given in Eqns. 4.3 - 4.5, and denoted as F_A, F_L and F_O, respectively,

$$F_A = \frac{1}{2}\iint (x_\xi^2 y_\eta^2 + x_\eta^2 y_\xi^2 - 2x_\xi x_\eta y_\xi y_\eta) d\xi d\eta \tag{4.3}$$

$$F_L = \frac{1}{2}\iint (x_\xi^2 + y_\xi^2 + x_\eta^2 + y_\eta^2) d\xi d\eta \tag{4.4}$$

$$F_O = \frac{1}{2}\iint (x_\xi^2 x_\eta^2 + 2x_\xi x_\eta y_\xi y_\eta + y_\xi^2 y_\eta^2) d\xi d\eta \tag{4.5}$$

Knupp & Steinberg (1993), Chibisov et al. (June 2006) and Khattri (2007) have shown that a linear combination of these functionals given by Eqn. 4.6 results in valid meshes for many engineering

applications.

$$F(x,y) = \omega_A F_A(x,y) + \omega_L F_L(x,y) + \omega_O F_O(x,y) \tag{4.6}$$

where the weight parameters ω_A, ω_L and ω_O verify the following conditions,

$$\omega_A + \omega_L + \omega_O = 1$$
$$\omega_A \geq 0,\ \omega_L \geq 0,\ \omega_O \geq 0$$

Applying the Euler-Lagrange relations to the weighted combination of the three functionals (Eqns. 4.6) yields the following system for a general mapping procedure,

$$\begin{aligned} \mathcal{L}'(x) &= a_1 x_{\xi\xi} + b y_{\xi\xi} + c_1 x_{\xi\eta} + d y_{\xi\eta} + e_1 x_{\eta\eta} + f y_{\eta\eta} = 0 \\ \mathcal{L}'(y) &= a_2 y_{\xi\xi} + b x_{\xi\xi} + c_2 y_{\xi\eta} + d x_{\xi\eta} + e_2 y_{\eta\eta} + f x_{\eta\eta} = 0 \end{aligned} \tag{4.7}$$

where,

$$a_1 = \omega_A y_\eta^2 + \omega_L + \omega_O x_\eta^2 \quad , \quad a_2 = \omega_A x_\eta^2 + \omega_L + \omega_O y_\eta^2$$
$$b = (\omega_O - \omega_A) x_\eta y_\eta$$
$$c_1 = 2(\omega_O - \omega_A) y_\xi y_\eta + 4\omega_O x_\xi x_\eta \quad , \quad c_2 = 2(\omega_O - \omega_A) x_\xi x_\eta + 4\omega_O y_\xi y_\eta$$
$$d = (\omega_O + \omega_A)(x_\xi y_\eta + x_\eta y_\xi)$$
$$e_1 = \omega_A y_\xi^2 + \omega_L + \omega_O x_\xi^2 \quad , \quad e_2 = \omega_A x_\xi^2 + \omega_L + \omega_O y_\xi^2$$
$$f = (\omega_O - \omega_A) x_\xi y_\xi$$

It is interesting to note that by taking the following combination, $\omega_A = 0, \omega_O = 0$ and $\omega_L = 1$, in the system of Eqns. 4.7 yields,

$$\begin{aligned} \mathcal{L}'(x) &= x_{\xi\xi} + x_{\eta\eta} = 0 \\ \mathcal{L}'(y) &= y_{\xi\xi} + y_{\eta\eta} = 0 \end{aligned} \tag{4.8}$$

This shows that the Euler-Lagrange equations for the length functional result in a linear and decoupled system of Laplace equations. The solution of Eqns. 4.8 leads to a smooth mesh by maintaining the

edges length in physical space but, for same practical applications, gives folded and degenerated cells around the concave regions of the boundary. Moreover, setting the weights for the area, $\omega_A = 1$, and the orhtogonality, $\omega_O = 1$, the functionals produce unfolded (valid) but nonsmooth grids. It is noted that the orthogonality functional fails to converge in some cases. Therefore, to overcome the resulting nonsmooth mesh and poor convergence behavior of using individual functionals, it was proposed to choose a combination of the area, length and orthogonality functionals. Generally, in this method, adjusting the weights requires a trial and error procedure for each new configuration to achieve a valid mesh.

In Knupp & Steinberg (1993), it is proposed that choosing the weighted parameters as $\omega_A = 0.5, \omega_O = 0.5$ and $\omega_L = 0$ gives valid results for a wide range of engineering applications. The resulting functional is called area-orthogonality or AO functional.

4.3 Ellipticity study of mapping operators

In variational methods, the ellipticity of the operator is the dominant factor that controls the characteristic of the final mesh. For instance, higher ellipticity makes the operators tend towards the length functional, where the boundary curvature has minimal effect on the final mesh.

The type of Eqns. 4.7 depends on the coefficient matrix $\mathcal{T}_{ij}$, defined as

$$\mathcal{T}_{11} = \begin{pmatrix} a_1 & b \\ b & a_2 \end{pmatrix}, \ \mathcal{T}_{12} = \begin{pmatrix} c_1 & d \\ d & c_2 \end{pmatrix}, \ \mathcal{T}_{22} = \begin{pmatrix} e_1 & f \\ f & e_2 \end{pmatrix} \tag{4.9}$$

Knupp & Steinberg (1993) have shown that the operator is elliptic in dimension k if

$$det(\sum_{i,j=1}^{k} \mathcal{T}_{i,j}\omega_i\omega_j) \geq c|\omega|^{2k} \tag{4.10}$$

By rewriting Eqn. 4.10 for two dimensional space gives,

$$det(\mathcal{T}_{11}\omega_1^2 + 2\mathcal{T}_{12}\omega_1\omega_2 + \mathcal{T}_{22}\omega_2^2) \geq c|\omega|^{2k} \tag{4.11}$$

for all vectors $\omega = (\omega_1, ..., \omega_k)$ and some constant c.

Applying this analysis to the Winslow operators, the coefficient matrix $\mathcal{T}_{ij}$ is defined as,

$$\begin{aligned} \mathcal{T}_{11} &= \begin{pmatrix} x_\eta^2 + y_\eta^2 & 0 \\ 0 & x_\eta^2 + y_\eta^2 \end{pmatrix} \\ \mathcal{T}_{12} &= \begin{pmatrix} -2(x_\xi x_\eta + y_\xi y_\eta) & 0 \\ 0 & -2(x_\xi x_\eta + y_\xi y_\eta) \end{pmatrix} \\ \mathcal{T}_{22} &= \begin{pmatrix} x_\xi^2 + y_\xi^2 & 0 \\ 0 & x_\xi^2 + y_\xi^2 \end{pmatrix} \end{aligned} \tag{4.12}$$

which always gives a positive value for Eqn. 4.11. This confirms that the Winslow operators are elliptic. The $\mathcal{T}_{ij}$ matrices for the length functional, where $\omega_L = 1$ in Eqn. 4.7 are,

$$\mathcal{T}_{11} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathcal{T}_{12} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad \mathcal{T}_{22} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.13}$$

Again, substituting these into Eqn. 4.11, yields

$$det(\mathcal{T}_{11}\omega_1^2 + 2\mathcal{T}_{12}\omega_1\omega_2 + \mathcal{T}_{22}\omega_2^2) = (\omega_1^2 + \omega_2^2)^2 \tag{4.14}$$

which is also always positive. Hence, the length functional is elliptic. Similarly, for the area functional, $\omega_A = 1$, $\mathcal{T}_{ij}$ can be written as

$$\begin{aligned} \mathcal{T}_{11} &= \begin{pmatrix} y_\eta^2 & -x_\eta y_\eta \\ -x_\eta y_\eta & x_\eta^2 \end{pmatrix} \\ \mathcal{T}_{12} &= \begin{pmatrix} -2y_\xi y_\eta & x_\xi y_\eta + x_\eta y_\xi \\ x_\xi y_\eta + x_\eta y_\xi & -2x_\xi x_\eta \end{pmatrix} \\ \mathcal{T}_{22} &= \begin{pmatrix} y_\xi^2 & -x_\xi y_\xi \\ -x_\xi y_\xi & x_\xi^2 \end{pmatrix} \end{aligned} \tag{4.15}$$

and by substituting these expresions into Eqn. 4.11, gives

$$det(\mathcal{T}_{11}\omega_1^2 + 2\mathcal{T}_{12}\omega_1\omega_2 + \mathcal{T}_{22}\omega_2^2) = 0 \tag{4.16}$$

Thus, in contrast to the length functional, the equations for the area functional are quasilinear and parabolic. This means that the solution to the area functional can be a discontinuous mesh in the physical domain and, from a mathematical point of view, may not exist for arbitrary boundary shapes. However, it has been the practical experience in this work that, the area functional gives a valid mesh for most applications. Similarly, The orthogonality operator, by taking $\omega_O = 1,\ \omega_L = \omega_A = 0$ in Eqn. 4.7, can be shown to be parabolic, as follows,

$$\begin{aligned}
\mathcal{T}_{11} &= \begin{pmatrix} x_\eta^2 & x_\eta y_\eta \\ x_\eta y_\eta & y_\eta^2 \end{pmatrix} \\
\mathcal{T}_{12} &= \begin{pmatrix} 4x_\xi x_\eta + 2y_\xi y_\eta & x_\xi y_\eta + x_\eta y_\xi \\ x_\xi y_\eta + x_\eta y_\xi & 4y_\xi y_\eta + 2x_\xi x_\eta \end{pmatrix} \\
\mathcal{T}_{22} &= \begin{pmatrix} x_\xi^2 & x_\xi y_\xi \\ x_\xi y_\xi & y_\xi^2 \end{pmatrix}
\end{aligned} \tag{4.17}$$

and, replacing $\mathcal{T}_{ij}$ in Eqn. 4.10, gives

$$det(\mathcal{T}_{11}\omega_1^2 + 2\mathcal{T}_{12}\omega_1\omega_2 + \mathcal{T}_{22}\omega_2^2) = 0 \tag{4.18}$$

Therefore, the orthogonality functional results in nonsmooth meshes for most of domains. Despite the fact that the value of discriminants for both area and orthogonality are equal zero, in Knupp & Steinberg (1993), the authors concluded that the use of orthogonality as a standalone method for automatically generating grids is not recommended. Examples will be presented in section 4.5 that clearly demonstrate the effect of the ellipticity of different individual and combinations of functionals.

4.4 Numerical discretization

The most straightforward method to solve the system of Eqns. 4.2 or Eqns. 4.7 is using a finite difference scheme on a structured parametric mesh. The Winslow operator, $\mathcal{L}$, and the variational operator, $\mathcal{L}'$, are significantly more complex to implement on unstructured meshes than on structured grids, but there are clear advantages in terms of flexibility and generality in using unstructured domain discretizations. However, little attention has been devoted to extend the method to unstructured meshes. The reason is that in contrast to Laplace and Poisson equations, the Winslow operator is in non-

conservative form, and therefore, conventional discretization schemes cannot be applied directly to unstructured meshes.

The extension to unstructured meshes will be investigated using two methods, one based on finite differences and the other on a finite volume technique to discretize Eqns. 4.2 and Eqns. 4.7.

4.4.1 Finite difference scheme

The idea of numerical discretization using finite difference schemes is to approximate the model equations, Eqns. 4.2 or Eqns. 4.7, by means of Taylor series expansions using discrete sets of points. To obtain a consistent scheme, there must be a sufficient number of neighbors to solve the resulting algebraic system resulting from these developments. For the classical structured grid arrangement, this technique is straightforward, but it is not generalizable to unstructured meshes because the number of cells surronding a given node is variable as illustrated in Fig. 4.2.

In Winslow (1997), a finite difference scheme is proposed for first and second derivatives on unstructured meshes with equilateral triangles where all the angles are equal to $\pi/3$. This was extended by Knupp (1999) to any N-valent logical mesh using a local mapping method which is created locally over each discrete polygon around a given node, (x_i, y_i). A computational stencil based on a regular polygon is created with the same number of corner points as the number of elements surrounding the node in computational space $\mathcal{C}$. From these, Taylor series expansions can be used to obtain finite

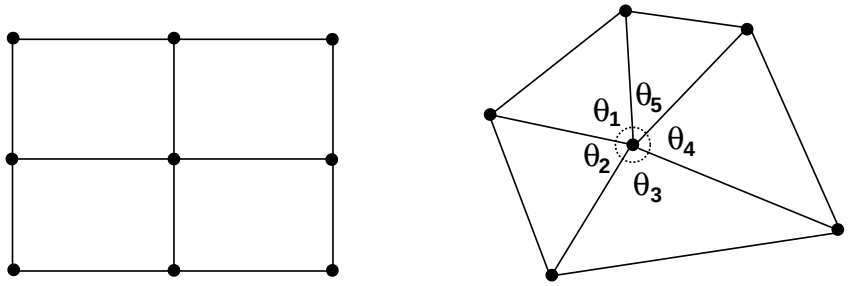


Figure 4.2 Structured and unstructured discretization stencils

difference expressions for the first and second order derivatives.

In this work, another finite difference method on unstructured meshes is introduced, which is easier to apply compared to the previous works and closer to the classical schemes used on structured grids. This algorithm is based on a 9-point Cartesian stencil formed inside each patch around the node (ξ_i, η_i) in the computational space, as shown in Fig. 4.3. The values at the stencil points can be obtained by interpolation or a least-squares reconstruction method. From these, the first and second order derivatives are approximated to second order accuracy on the Cartesian stencil with an equal spacing $\Delta\xi_i = \Delta\eta_i$. In the present study, the values of the dependent variables on the stencil nodes are linearly interpolated from the nodal values of the elements surrounding the node, and $\Delta\xi_i, \Delta\eta_i$ are chosen as a fixed fraction of the length of the shortest edge connected to node (ξ_i, η_i).

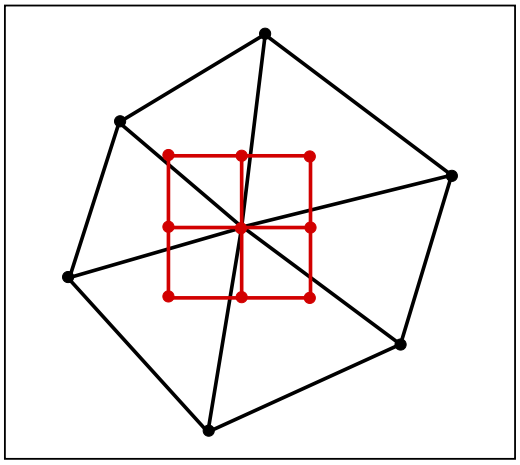


Figure 4.3 Proposed 9 point stencil for the finite difference discretization of the mapping operation for unstructured mesh

The discrete operators for $\mathcal{L}(x)$ and $\mathcal{L}(y)$ are solved using an SOR-type iterative procedure. Since, almost all cells generated in the computational space satisfy the Delaunay criteria, the polygons are very close to the ideal regular polygon presented in Knupp (1999). This property insures that all the information from the polygon surrounding the node contributes to the solution process. Fig.4.4 illustrates a situation where a triangle, $\triangle MPN$, does not contribute directly to the construction of the stencil's model. This could give rise to an uneven contribution or skewed weighting of the nodal information. However, it is estimated that this would, heuristically, be smoothed out by the neighboring stencils.

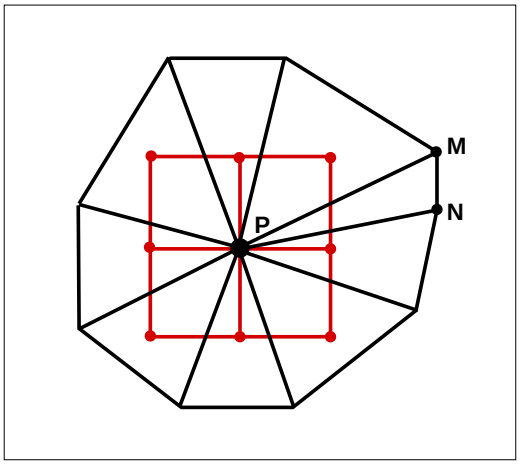


Figure 4.4 Missing information by cell(M,P,N) on the 9 point stencil

4.4.2 Finite volume scheme

Recently, Karman (2010) has introduced a finite volume descretization of the Winslow operator based on linearizing the equations by freezing the metric coefficients of the equations. For a given point, a control volume is constructed in virtual space where the element shapes are nearly ideal (as equilateral triangles and quadrilaterals in hybrid meshes). These virtual control volumes are created locally around each node in Ω as a local computational space with the same number of neighboring nodes as in physical space (Knupp (1999)). Several advantages are enumerated in Karman (2010) to using such virtual control volumes. Firstly, no existing valid computational mesh is required and all that is needed is the element connectivity. The second advantage is using the equilateral triangles around each node which influence the quality of the resulting mesh in the physical space. This extension to the control volume scheme is now described in detail.

As mentioned in the previous section, the Winslow operators are in nonconservative form because the three coefficients, $g11, g12$ and $g22$ are functions of gradients of the dependent variables in the computational space. Using a linearization procedure, Eqns. 4.2 can be integrated 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. 4.5. The cell edges divide each triangular element into three equal areas, and, collectively, these areas form

nonoverlapping contiguous control volumes associated with a node in the mesh. The hashed region in Fig. 4.5 indicates a control volume with a centroid node which is the storage location of all dependent variables.

This results in the integral form of the linearized Eqns. 4.2

$$g_{11}\iint x_{\xi\xi}d\Gamma - 2g_{12}\iint x_{\xi\eta}d\Gamma + g_{22}\iint x_{\eta\eta}d\Gamma = 0 \tag{4.19}$$

Applying the divergence theorem to the second order derivative terms, for example for the first one, gives

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

where the components of function F is $F = (x_\xi, 0)$. A similar procedure is applied to the $x_{\eta\eta}$ term. One critical step in the procedure is the calculation of the cross derivatives which requires a special treatment. In Karman (2010), the authors proposed the use of augmented cells around the control volumes. In the present work, it is proposed to solve all the operators in computational domain and without considering the virtual control volumes described in Karman (2010). This method uses actual control volume in computational space leading to a simpler arithmetic procedure. In addition, the cross derivatives are found using the same control volume and without considering auxiliary cells.

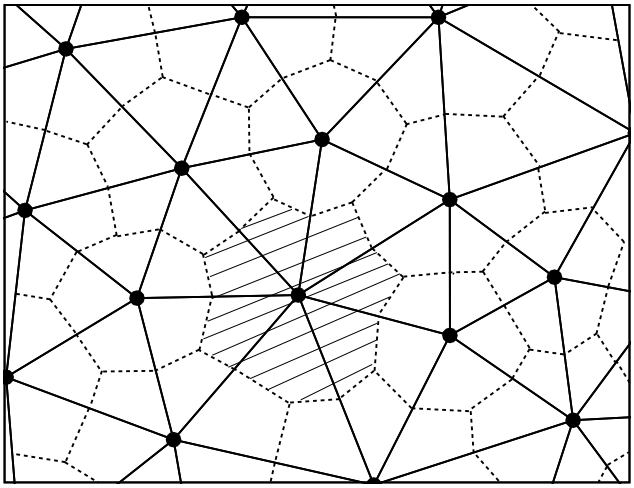


Figure 4.5 Computational mesh and a control volume

Applying this procedure to the term $x_{\xi\eta}$

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

This can be computed using different techniques depending on the evaluation of the term $\bigtriangledown \cdot Q$. In Karman (2010), an expanded stencil, shown by the dashed line in Fig. 4.6, is chosen to evaluate the cross derivative terms in order to bring more information to the centroid point of each patch.

In this work, the same control volume shown in Fig. 4.5 is used to compute the cross derivative terms and by integrating the terms in both ξ and η directions, the value of fluxes in these two directions will be obtained. The average value of these two fluxes gives the net flux of that cross derivative term,

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

where $Q_1 = (0, x_\eta)$ and $Q_2 = (x_\xi, 0)$.

Integrating over the control volume and applying the divergence theorem for each dependent variable, for example ξ, gives

$$\iint \bigtriangledown \cdot F d\Gamma = \oint F.\hat{n} dS \tag{4.20}$$

The term under the RHS integral represents the net flux that passes through the surface of the volume

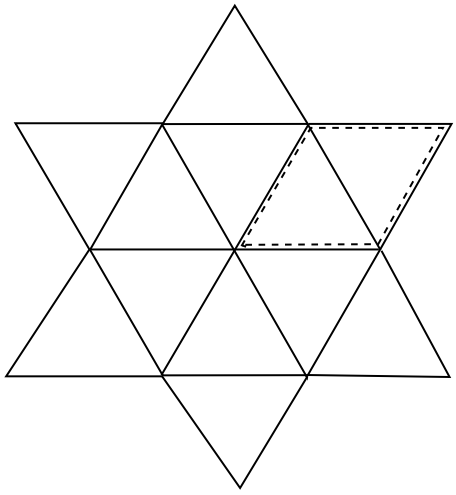


Figure 4.6 Augmented stencil used in Karman (2010) for cross derivative terms

and, for the Winslow operator, can be evaluated as

$$g_{11}\oint x_\xi n_x dS - 2g_{12}[\frac{1}{2}(\oint x_\eta n_x dS + \oint x_\xi n_y dS)] + g_{22}\oint x_\eta n_y dS = 0 \qquad (4.21)$$

$$g_{11}\oint y_\xi n_x dS - 2g_{12}[\frac{1}{2}(\oint y_\eta n_x dS + \oint y_\xi n_y dS)] + g_{22}\oint y_\eta n_y dS = 0 \qquad (4.22)$$

where, following a counter clockwise direction, the lengths of the sides of each control volume are calculated by

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

It has been our specific experience that for the cross derivative terms, applying the Green's theorem only to one component on each actual control volume side around node (ξ_i, η_i) yields a degenerated final mesh in most cases for 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, wrongly deforms the final mesh.

4.5 Validation of the mapping methodology

The proposed discretizations, based on both finite difference and the finite volume techniques, have been validated using several domain configurations presenting a gradation of geometric difficulties. A central difference method is used for the modified finite difference technique and the variables are updated using gauss seidel iterative method in the finite volume approach. The comparison between discretization schemes and mapping operators are presented separately in the following sections.

4.5.1 Validation of the finite difference scheme

First of all, to demonstrate the validity and accuracy of the finite difference technique with a 9-point stencil shown in Fig. 4.3, a sequence of mapping operations were carried out. This consisted of mapping a physical domain to a computational domain, $\Omega \rightarrow \mathcal{C}$ using Laplace's equations, followed by a re-mapping of the resulting mesh from the computational to the original physical domain, $\mathcal{C} \rightarrow \Omega$, by applying the Winslow operators. This procedure is shown schematically in Fig. 4.7.

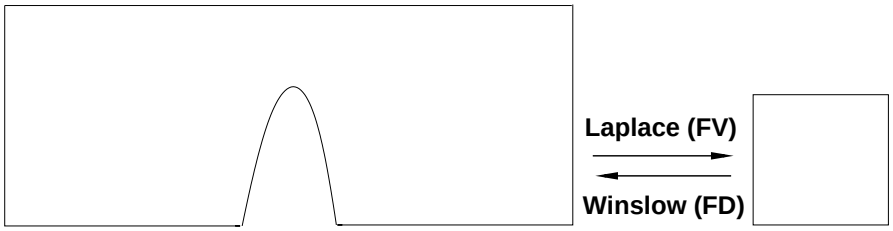


Figure 4.7 Solving Laplace's equation to transfer physical mesh to computational domain using a finite volume discretization technique, and solving Winslow equation on the computational mesh using the modified finite difference scheme to recover physical space by the computational mesh.

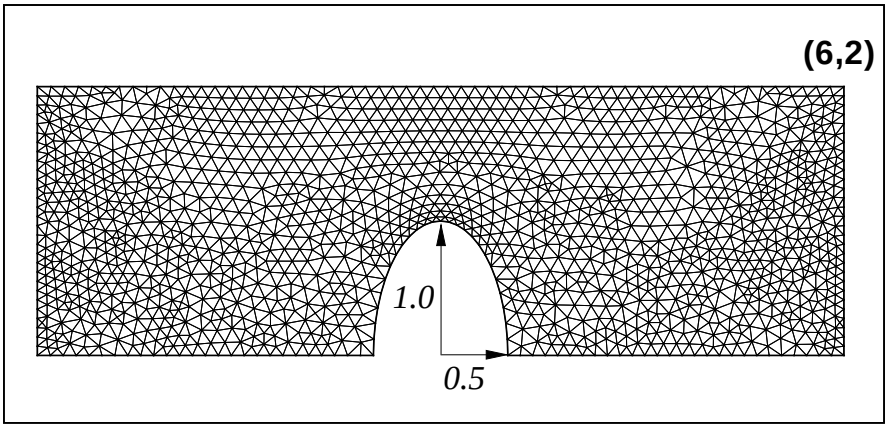
By comparing the original mesh in physical space with the recovered mesh obtained by the finite difference scheme, the accuracy of the method can be evaluated.

This cyclic sequence was carried with an isotropic mesh generated around a 200% bump shown in Fig. 4.8(a). Solving Laplace's equations on physical space Ω, based on a finite volume discretization method, the mesh in computational space is obtained as shown in Fig. 4.8(b). The effect of the bump is clearly reflected in the computational domain.

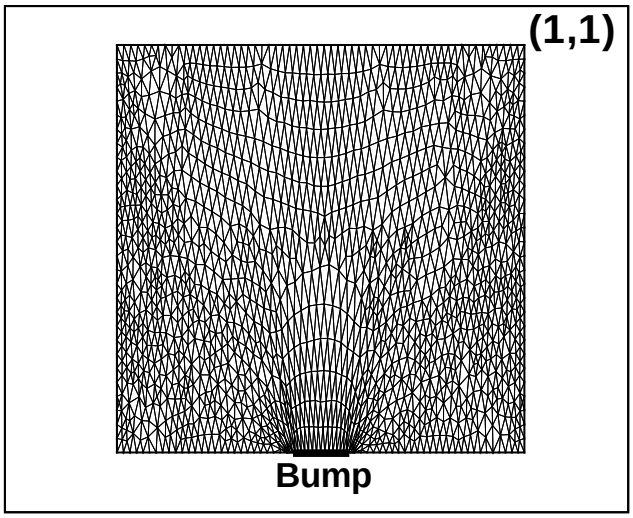
The Winslow operators are then applied on the mesh shown in Fig. 4.8(b), using the proposed 9-point cartesian stencil. The boundary condition in this step is taken as the physical boundary shown in Fig. 4.8(a).

Comparing the original mesh in the physical domain, Fig. 4.8(a), and the final recovered mesh shown in Fig. 4.9, gives an error which is shown in Fig. 4.10. The error is computed as the difference between the original and the resulting physical nodal coordinates (Error$= |X_{original\ mesh} - X_{FD}|$) and it can be observed from Fig. 4.10 that these are below 0.8%. This example confirms the ability of the method to transfer the coordinates of the grid points from computational to the physical space. It should be

noted that the resulting mesh obtained by the finite difference scheme is not relevant to the size of the defined stencil inside the patches, which illustrates the consistency of the method.



(a) Generated mesh in physical domain



(b) Mesh in computational space

Figure 4.8 Mapping from physical to computational space

To further illustrate this discretization scheme, a straight line, with zero thickness and coinciding nodes on both sides, is mapped to a circle using the Winslow operator. As can be seen in Fig. 4.11(b), all cells are valid around the circle.

4.5.2 Comparison of finite volume and finite difference schemes

To compare the finite difference and finite volume discretizations of the Winslow operator, the mapping of a mesh around a slit in the (ξ, η) plane to a sharp wedge in Ω was used. Fig. 4.12(b) shows

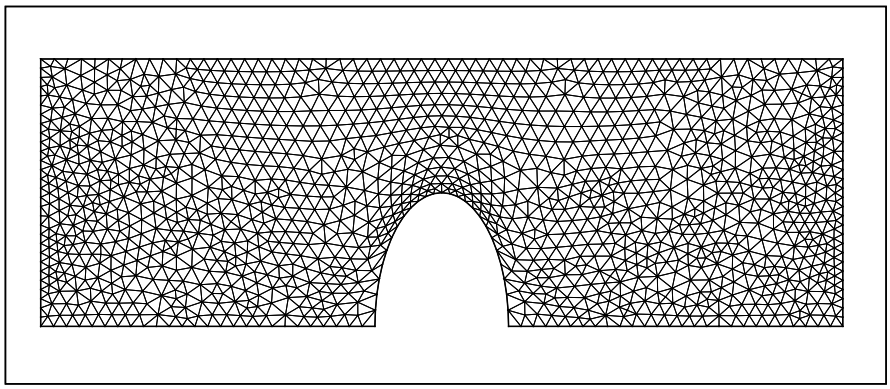


Figure 4.9 Final mesh in physical domain

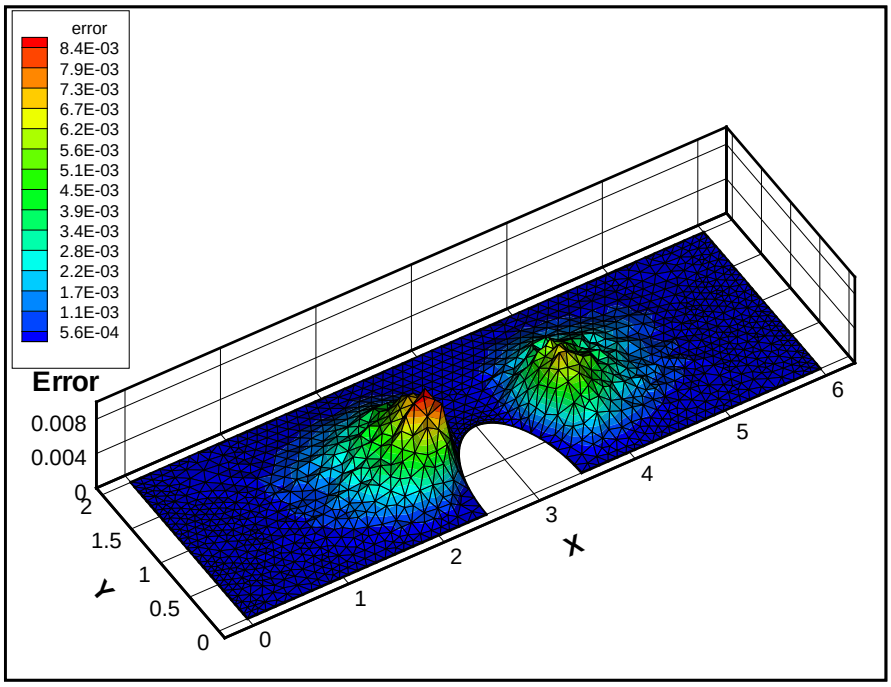
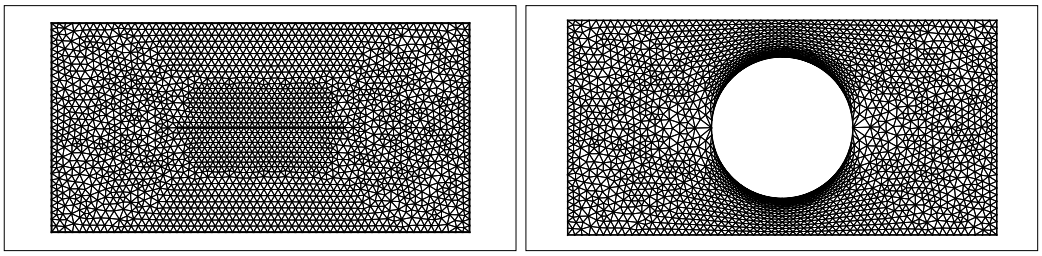


Figure 4.10 The error value at each node indicates the difference between the original and the resulting mesh obtained by finite difference scheme

the resulting mesh obtained using the finite volume discretization. This test case has also been solved using the finite difference scheme to illustrate the limitations of this discretization method. While the finite difference scheme is highly efficient for configurations with mild boundary curvature variations, the degenerate grids in Fig. 4.12(c) clearly show its limits when large curvature variations, nearing discontinuities, are presented.

The next example exhibits a comparison of convergence histories of the computations performed



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

Figure 4.11 Mapping a straight line with coincided nodes to a circle using Winslow equations and finite difference scheme

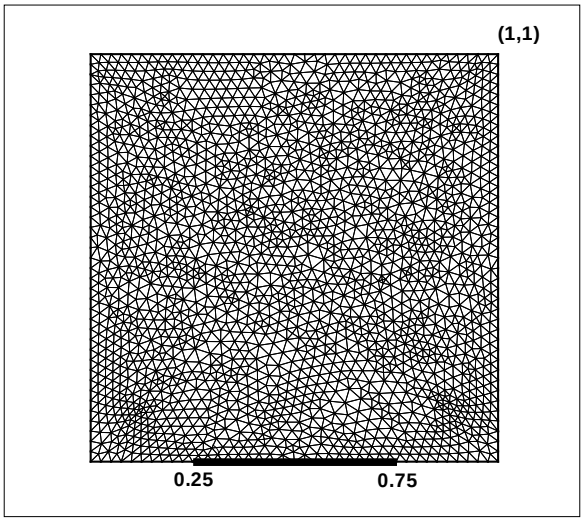
by both discretization techniques for the computational mesh shown in Fig. 4.11(a). The resulting meshes are shown in Figs. 4.13(a) and (b) and the convergence rate of computations are presented in Fig. 4.13(c). The cost of unit calculation using finite difference scheme is 2 times more than the finite volume method, but as it can be seen by Fig. 4.13(c) this method converges around 10 times faster than finite volume technique. Hence, the global efficiency of the finite difference scheme is 5 times more than the finite volume technique. The overall performance is comparable in terms of memory.

4.5.3 Validation and comparison of different mapping operators

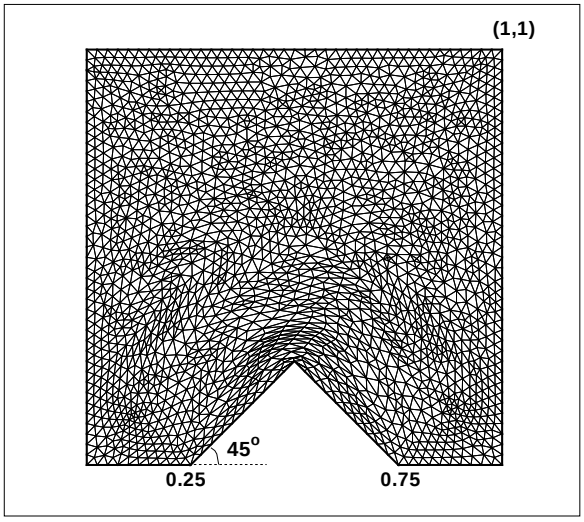
To compare the two mapping operators, Eqns. 4.2 (Winslow) and 4.7 (functionals), a generic mesh around a circle in computational space, Fig. 4.14(a), is mapped to a configuration around a four-petal rose shown in Fig. 4.14(b),(c), using finite volume discretization technique.

As it can be seen from Fig. 4.14(b), the cells near the corners maintained their isotropy by solving Eqns. 4.7 with $\omega_A = \omega_O = 0.5, \omega_L = 0$, while the Winslow operator, Eqns. 4.2, cannot achieve the same quality around highly curved regions as shown in Fig. 4.14(c). The closeup part of Fig. 4.14(c) illustrates this effect as the highest curvature part is smeared rapidly towards the interior cells when using the Winslow operator, whereas in Fig. 4.14(b) the cells are pulled properly towards the corners. A formal comparison of the grid quality based on the minimum angle is shown in Figs. 4.15. These angle distributions show that the AO functional operator produces fewer skewed cells than the Winslow operator.

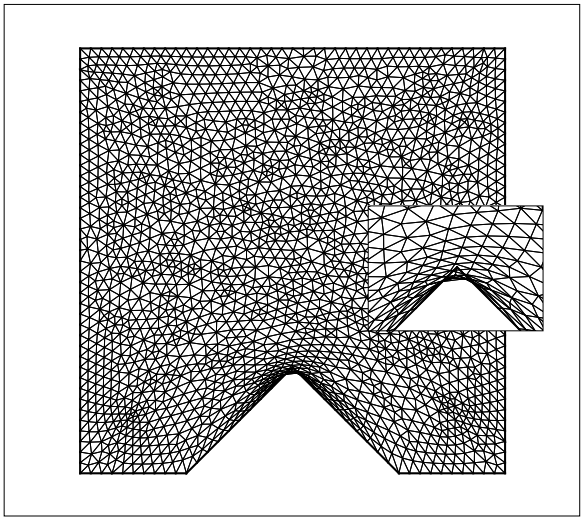
Individual weighted functionals and weighted combinations of length (L), area (A) and orthogonality (O) are studied to distinguish the effect of each, using the discontinuous spike geometry. As can



(a) Grid in computational space

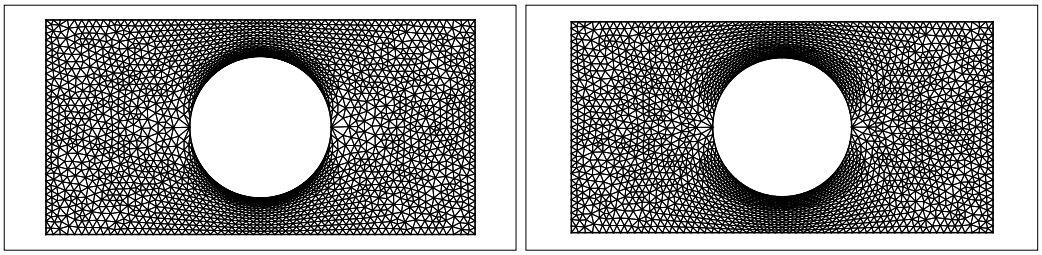


(b) Finite volume mesh

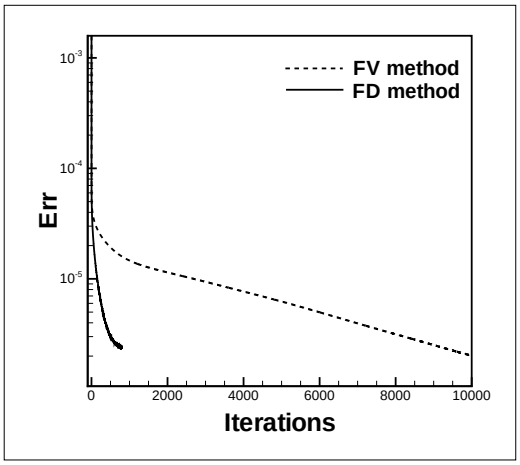


(c) Finite difference mesh

Figure 4.12 Mapping of a slit in computational domain to a spike in physical space using Winslow equations

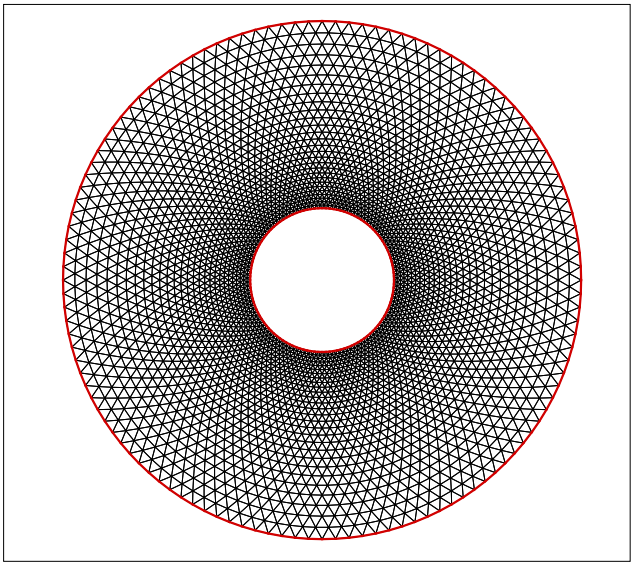


(a) Finite difference scheme (b) Finite volume scheme

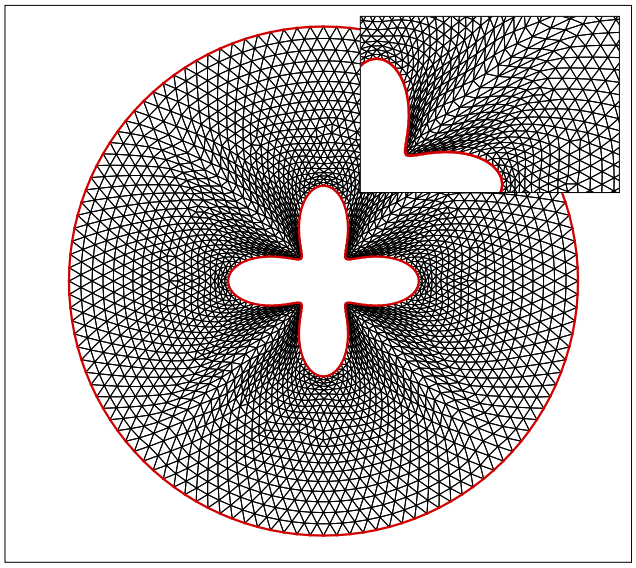


(c) Convergence history

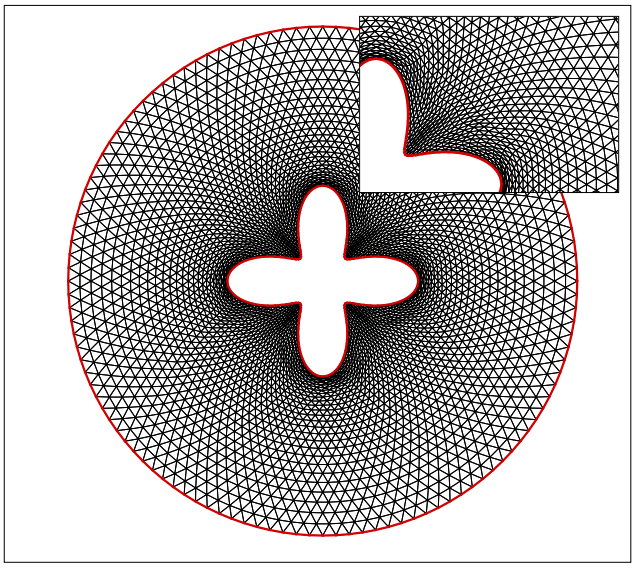
Figure 4.13 A straight line with coinciding nodes mapped to a circle using the Winslow operator



(a) Mesh around a generic multiply connected domain: a circle in computational space

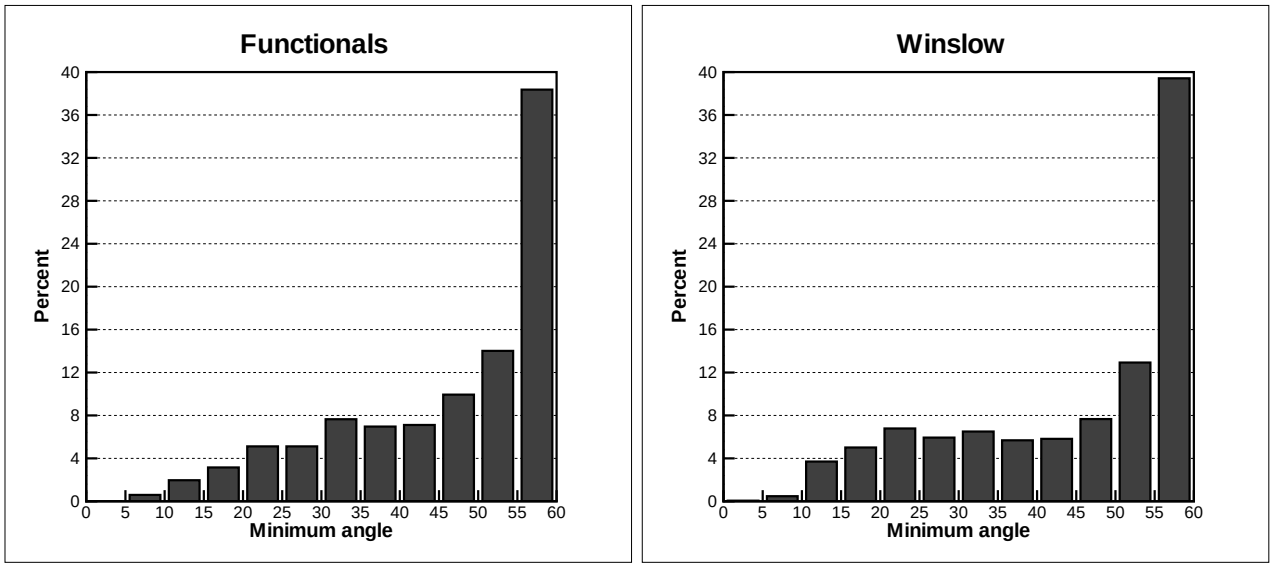


(b) Mesh using the AO functional operator



(c) Mesh using the Winslow operator

Figure 4.14 Comparison of the two mapping operators



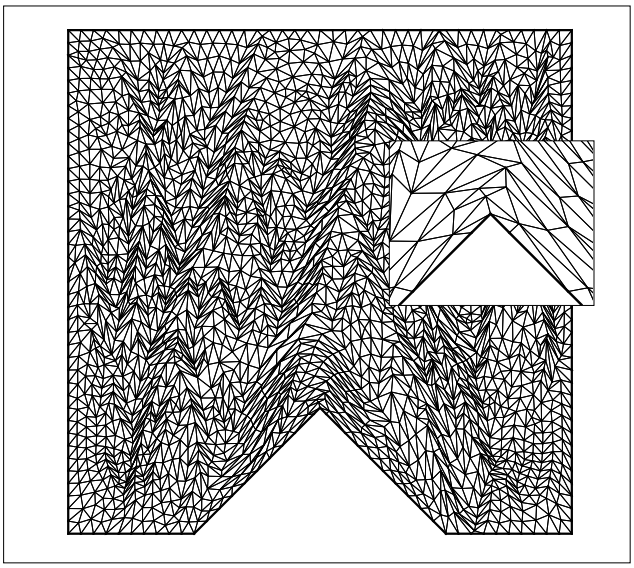
(a) For the AO functional operator (b) For the Winslow operator

Figure 4.15 Comparison of the mesh quality for the two mapping operators using Minimum angle distribution

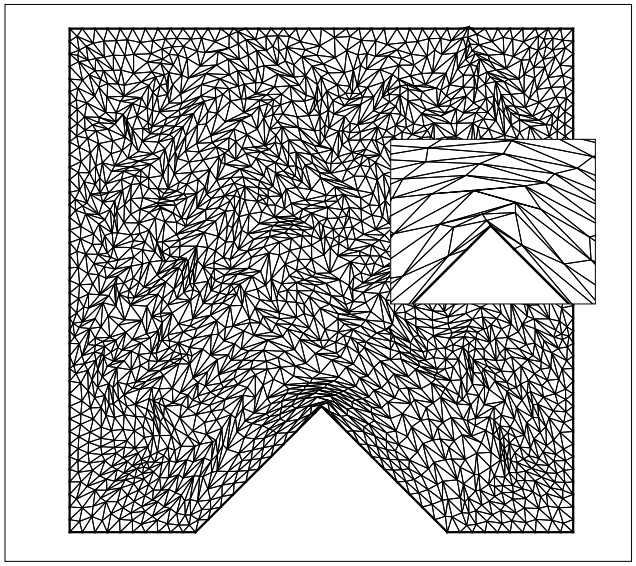
be seen in Figs. 4.16(a) and (b), the area and orthogonality functionals, when used alone, create nonsmooth meshes due to lack of ellipticity of their corresponding Euler-Lagrange equations, whereas the length functional, Fig. 4.16(c), gives a smooth mesh regardless of the boundary curvature, but in some cases generate invalid cells. The area-length (AL) combination, $\omega_A = \omega_L = 0.5$ and $\omega_O = 0$, tries to overcome two important limitations of its individual functionals, namely lack of smoothness for the area and folding for the length functional. However, the solution to the AL equations fails around the tip of the spike, as shown in Fig. 4.17(a). The same effect can be seen in Fig. 4.17(b) for orthogonality-length (OL), $\omega_O = \omega_L = 0.5$ and $\omega_A = 0$, due to effect of the length functional. But here, more cells are folded compared to Fig. 4.17(a). As it is shown in Fig. 4.17(c) even though the AO meshes are not always completely satisfactory because of its nonelliptic operator, the grids smoothness is better than its individual functionals.

4.6 Mesh smoothness

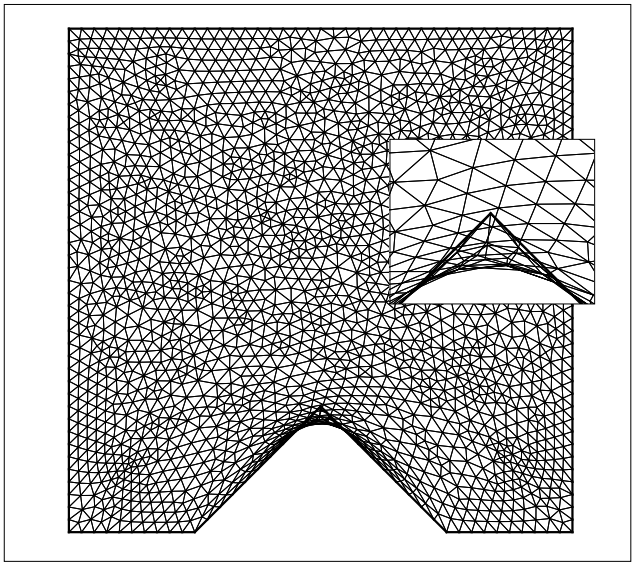
In contrast to the traditional definition of mesh quality, which usually considers individual criteria of each element, smoothness of a mesh has a global definition. Thus, these two distinct definitions of mesh quality and mesh smoothness may be contradictory for some cases. Indeed, a smoother mesh



(a) A, $\omega_A = 1, \omega_O = \omega_L = 0$

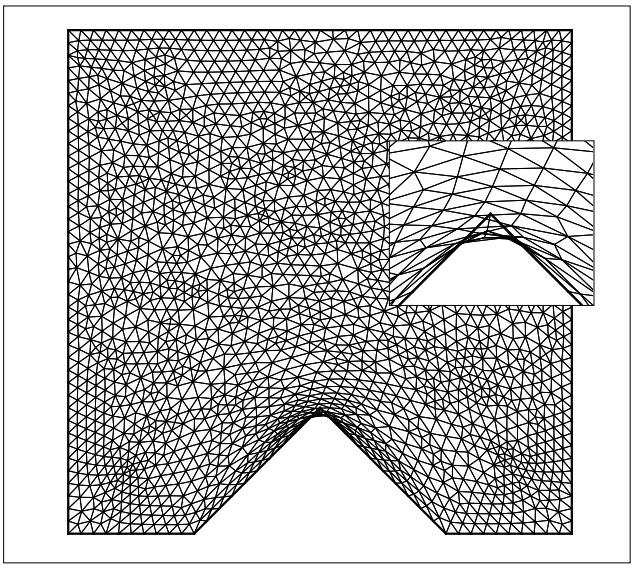


(b) O, $\omega_O = 1, \omega_A = \omega_L = 0$

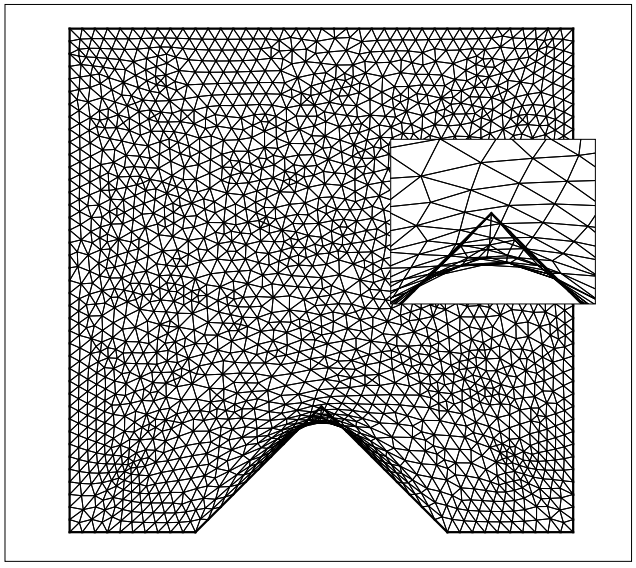


(c) L, $\omega_L = 1, \omega_A = \omega_O = 0$

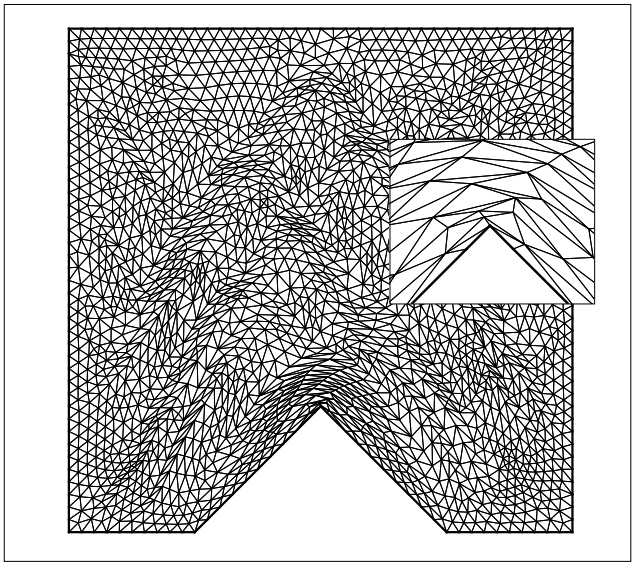
Figure 4.16 Comparison of individual functionals for a spike geometry



(a) AL, $\omega_O = 0, \omega_A = \omega_L = 0.5$



(b) OL, $\omega_A = 0, \omega_L = \omega_O = 0.5$



(c) AO, $\omega_L = 0, \omega_A = \omega_O = 0.5$

Figure 4.17 Comparison of combined functionals for a spike geometry

does not necessarily imply better mesh quality.

In this work, a new method, based on the local area ratio, is presented in order to compare the smoothness of the resulting meshes obtained by different maping operators. The mesh smoothness is quantified for each cell as

$$SR_i = \frac{A_i}{max(A_n)} \quad , \quad for\ n = 1, 2, 3 \tag{4.23}$$

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. The best values for SR_i is as close as possible to one.
The Smoothness Factor (SF) of a mesh, is defined as follows,

$$SF = \frac{1}{N_e} \sum_{1}^{N_e} min(SR_i, \frac{1}{SR_i}) \tag{4.24}$$

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.

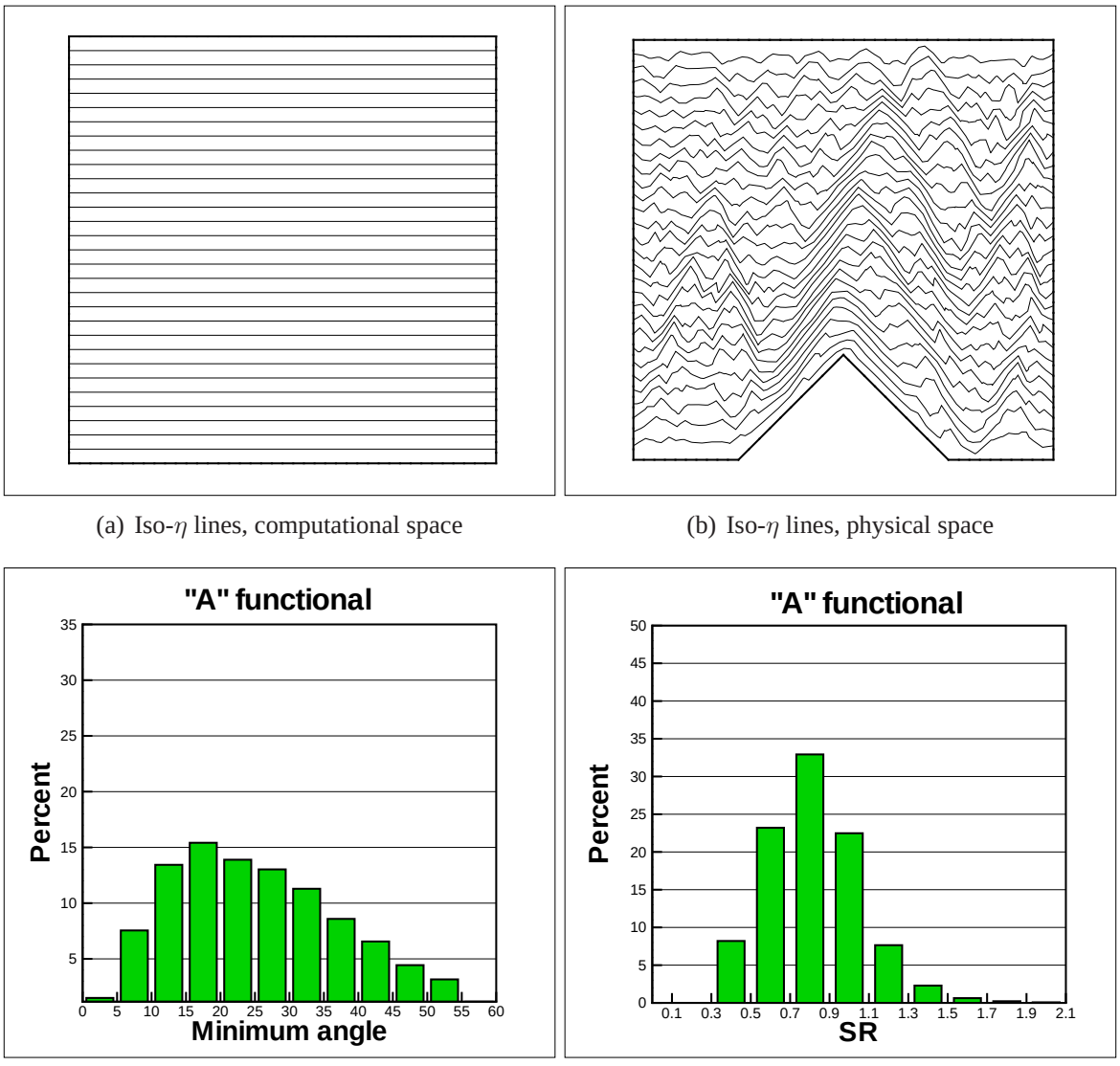
In addition to the SR_i distribution and the global value of SF for a mesh, another measure to assess the smoothness of a mesh by each mapping operator is introduced. As it is known, in computational space, the iso-η (and iso-ξ) values are straight lines and the effect of the mapping operators is to deflect these straight lines according to the defined boundary conditions in physical space. Some operators distort these lines in a smooth manner while some others do not. For a better understanding of the properties of these mapping operators and their effects on the final mesh, these two criteria are compared for a given geometry.

Fig. 4.18(a),(b) show the iso-η lines in computational space and their deflection in the $(x(\xi, \eta), y(\xi, \eta))$ space by applying the mapping operator based on the area functional. As mentioned earlier, the resulting nonsmooth mesh or iso-η (iso-ξ) lines in physical domain can be related to the lack of ellipticity of this operator. The poor quality of the generated mesh by the area functional is quantified in Fig. 4.18(c), while Fig. 4.18(d) shows the area ratio given by Eqn. 4.23 for all cells. A distribution of SR close to 1 represents a smoother mesh, which, for this specific example gives only 23% of the cells close to $SR \approx 1$.
Fig. 4.19(a) shows the effect of another nonelliptic operator on the deformation of iso-η lines, with $\omega_O = 1$ and $\omega_L = \omega_A = 0$. In this case, despite the increase in mesh quality compared to the area

functional, the smoothness of the mesh decreases slightly. As expected and shown in Fig. 4.20, the AO functional gives a better quality and a smoother mesh but, the best quality and smoothest mesh is obtained by the Winslow operators as depicted in Fig. 4.21.

Table 4.1 summarizes these results by a comparison of the SF for these operators. The Winslow operator yields the maximum value, followed by AO and A functionals. The orthogonality, O, operator generates the least smooth mesh, with the minimum SF value in this table.

To further assess their characteristics, these operators were applied to a four-petal geometry. As



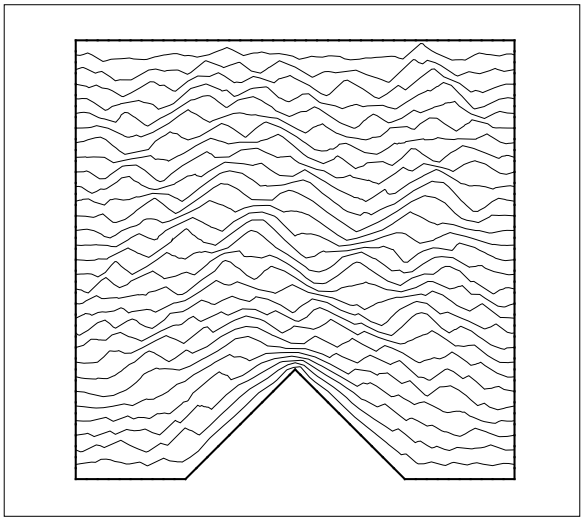
(a) Iso-η lines, computational space

(b) Iso-η lines, physical space

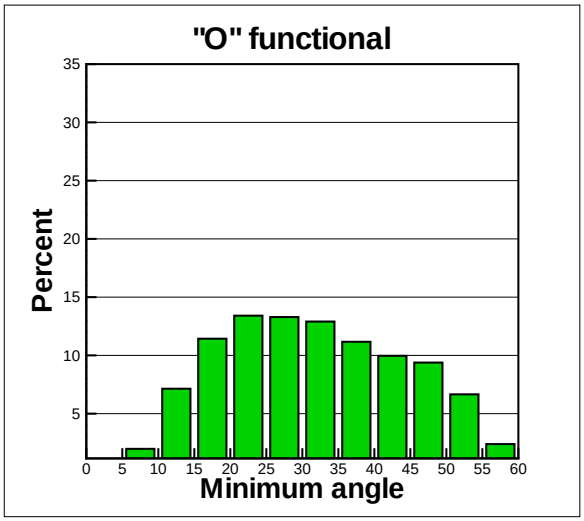
(c) Mesh quality

(d) Mesh smoothness

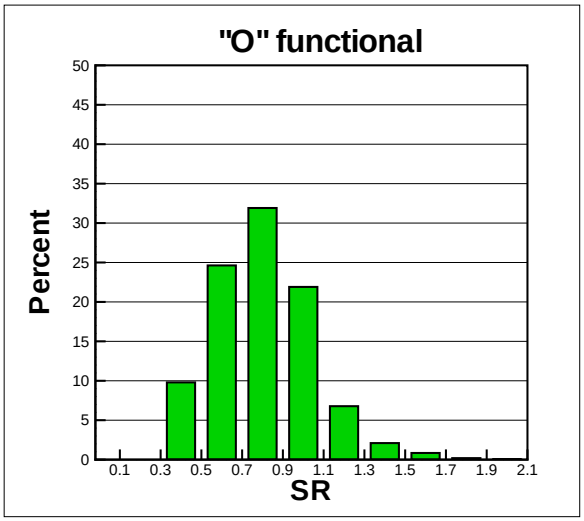
Figure 4.18 Comparison of mesh quality and mesh smoothness by 'A' functional where $SF = 0.746$



(a) Iso-η lines, physical space

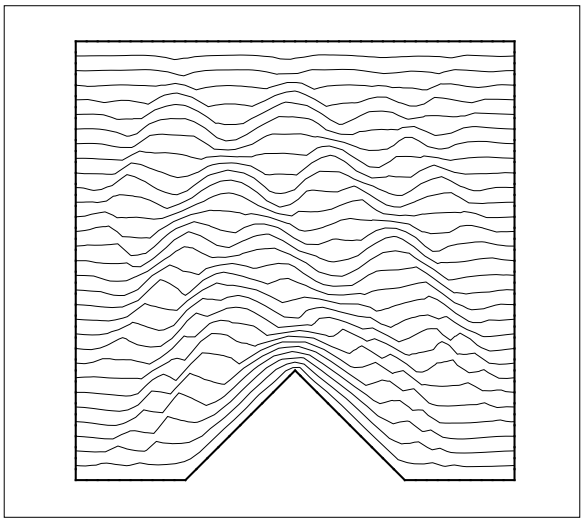


(b) Mesh quality

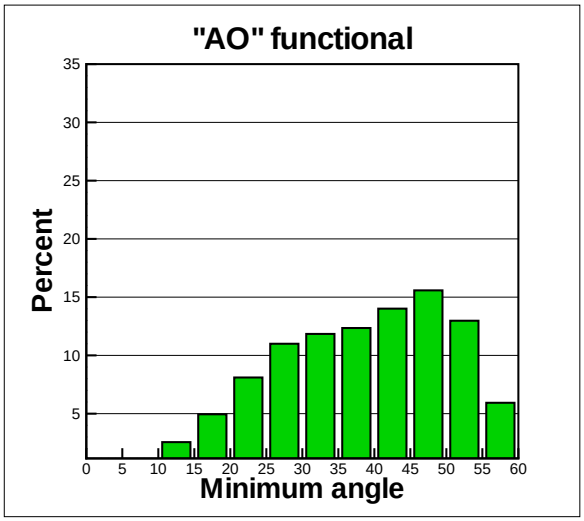


(c) Mesh smoothness

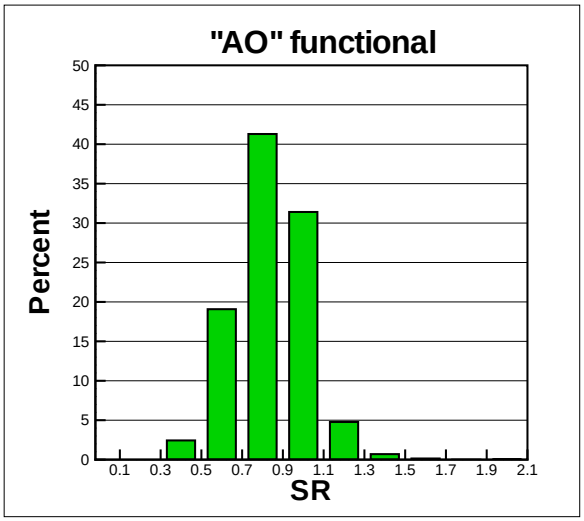
Figure 4.19 Comparison of mesh quality and mesh smoothness by 'O' functional where $SF = 0.739$



(a) Iso-η lines, physical space

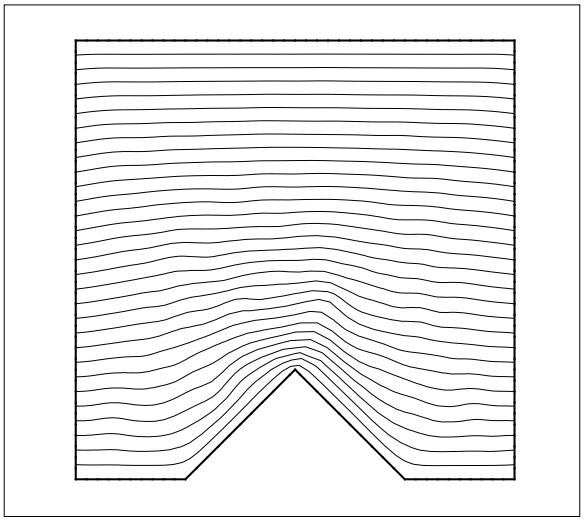


(b) Mesh quality



(c) Mesh smoothness

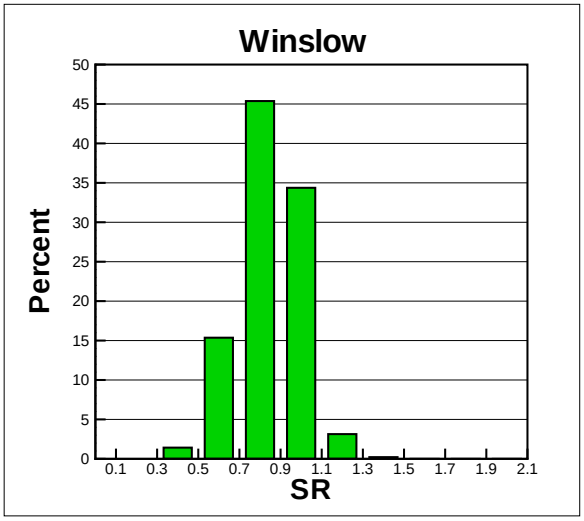
Figure 4.20 Comparison of mesh quality and mesh smoothness by 'AO' functional where $SF = 0.817$



(a) Iso-η lines, physical space



(b) Mesh quality



(c) Mesh smoothness

Figure 4.21 Comparison of mesh quality and mesh smoothness by Winslow operators where $SF = 0.824$

Table 4.1 Comparison of global Smoothness Factor, "SF", for the spike geometry, based on different mapping operators

Operator	Smoothness Factor
"Winslow" Fig. 4.12 (b)	0.824
"A" Fig. 4.16 (a)	0.746
"O" Fig. 4.16 (b)	0.739
"AO" Fig. 4.16 (f)	0.817

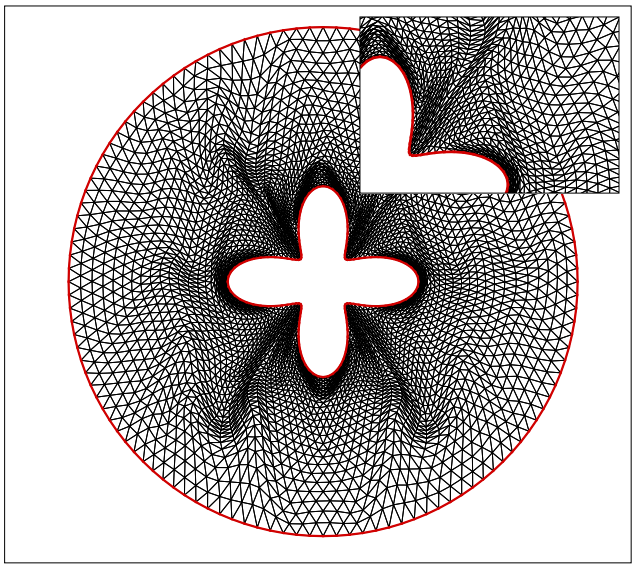
expected and shown in Figs. 4.22(a) and 4.22(b), the mesh obtained using the area and orthogonality functionals are discontinuous, but are valid. Fig. 4.22(c) illustrates again that the length functional, or Laplace's equation, creates a very smooth grid around the physical boundaries, however folded cells appear in nonconvex regions. Results for combined functionals, AL, OL and AO operators are shown in Fig. 4.23. The length functional inverts the cells in AL and OL operators, even though the whole mesh is smooth. However, the nonelliptic AO operator gives a satisfactory and valid mesh for this specific geometry. The Smoothness Factor, SF, for all valid meshes around the four-petal rose are shown in Table 4.2. As expected from these various operators, the Winslow operator generates the smoothest mesh, followed by results for the AO operator, while the smoothness of the meshes obtained using A and O functionals are very close to each other.

4.7 Limitations

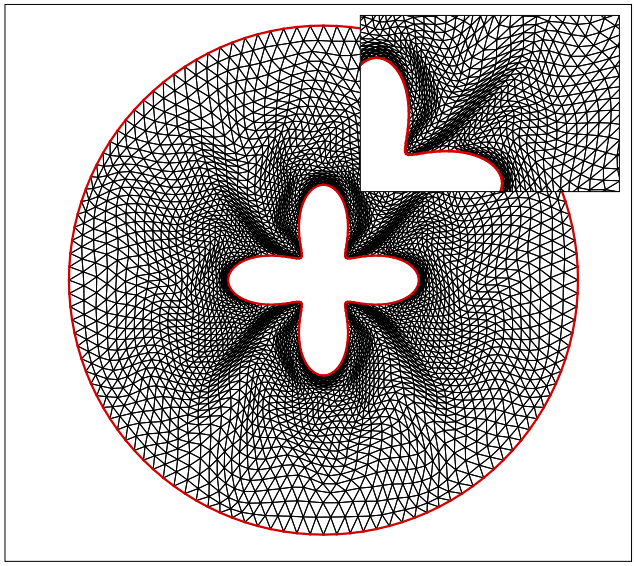
The objective of analyzing the mapping operators is to develop a procedure within the global methodology for moving grids. As such, the choice of a particular technique should verify the following

Table 4.2 Comparison of global Smoothness Factor, "SF", for the four-petal geometry, based on different mapping operators

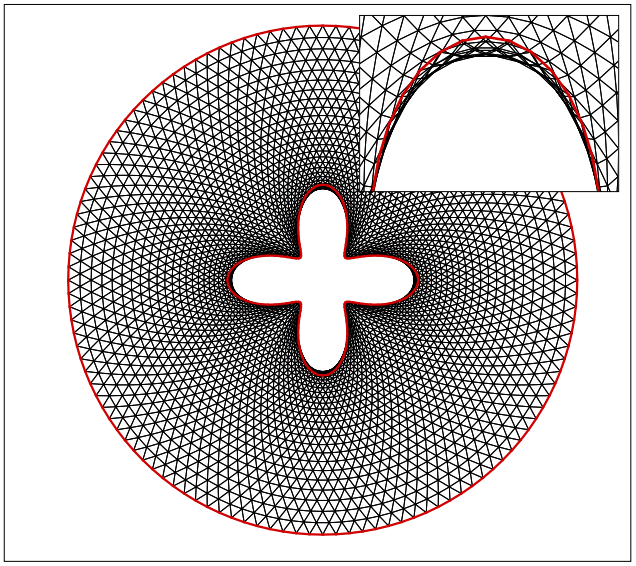
Operator	Smoothness Factor
"Winslow" Fig. 4.14 (c)	0.938
"A" Fig. 4.22(a)	0.779
"O" Fig. 4.22(b)	0.774
"AO" Fig. 4.23(c)	0.828



(a) A, $\omega_A = 1, \omega_O = \omega_L = 0$

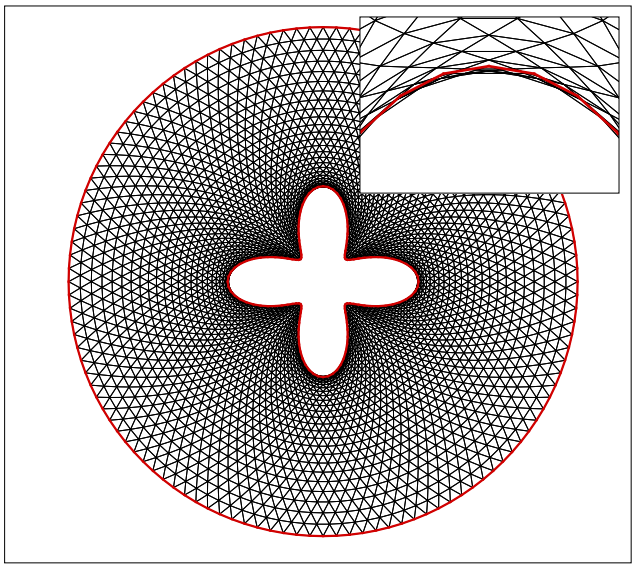


(b) O, $\omega_O = 1, \omega_A = \omega_L = 0$

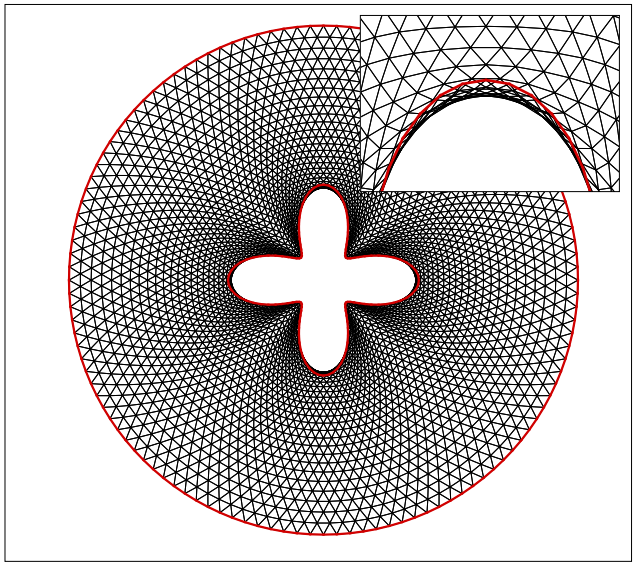


(c) L, $\omega_L = 1, \omega_A = \omega_O = 0$

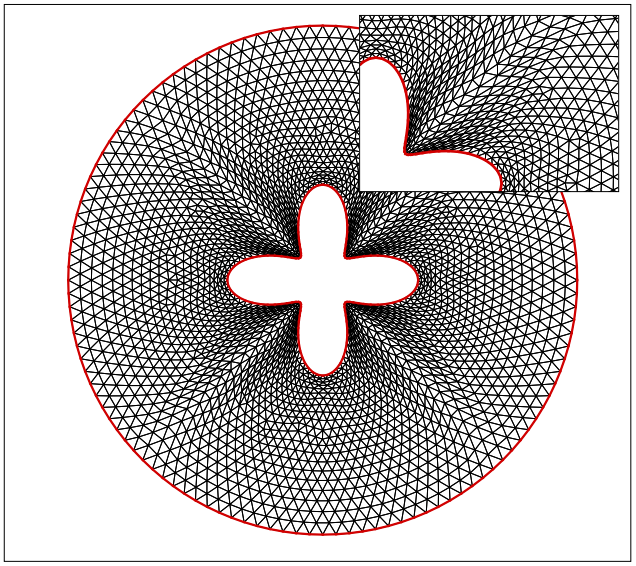
Figure 4.22 Comparison of individual functionals for a four-petal rose geometry



(a) AL, $\omega_O = 0, \omega_A = \omega_L = 0.5$



(b) OL, $\omega_A = 0, \omega_O = \omega_L = 0.5$



(c) AO, $\omega_L = 0, \omega_A = \omega_L = 0.5$

Figure 4.23 Comparison of combined functionals for a four-petal rose geometry

requirements:

- produce valid or invertible mesh,
- have good smoothness or quality,
- be well adapted to the physical boundaries.

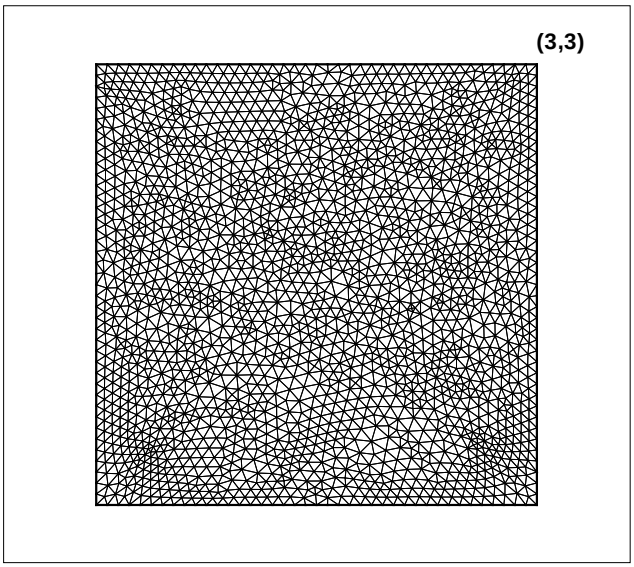
To asses possible candidates with respect to these criteria, different functional and the Winslow operators have been applied to various geometries, and the resulting meshes compared for quality and smoothness. To evaluate the behavior of the mapping operators for extreme cases, two different boundaries have been chosen. In the first example, the operators deal with a highly deformed boundary, whereas in the second one, the effect of a discontinuity on a nontrivial boundary has been considered.

Figs. 4.24(a)-(c) depict the computational mesh and two different physical boundaries, a high amplitude bump and a large spike. As it can be seen from Figs. 4.25(a)-(f), amongst the six different mapping operators applied on the computational mesh with the boundary condition shown in Fig. 4.24(b), only the A, AO functionals and the Winslow operators generate valid meshes in physical space. In this example, the O functional generates an invalid and nonsmooth mesh, while the role of the Length parameter in the AL and OL functionals is clearly shown in Figs. 4.25(c) and (d). Moreover, as it can be guessed from the type of the mapping operator, the Winslow equations, which are elliptic, naturally diffuse the effect of the bump towards the interior of the domain. On the other hand, the effect of the bump in Figs. 4.25(a) and (e) is limited mostly around its peak where applying the nonelliptic A and AO operators. In Figs. 4.25(a) and (e), the resulting meshes exhibits characteristics related to the type of the operators, that is parabolic for the A functional compared to the AO functional which is hyperbolic. From a mathematical point of view, it is known that the parabolic equations are closer to the elliptic than the hyperbolic counter part. Therefore, deformed cells are confined to a region closer to the peak of the bump in Fig. 4.25(e), when compared to the Fig. 4.25(a).

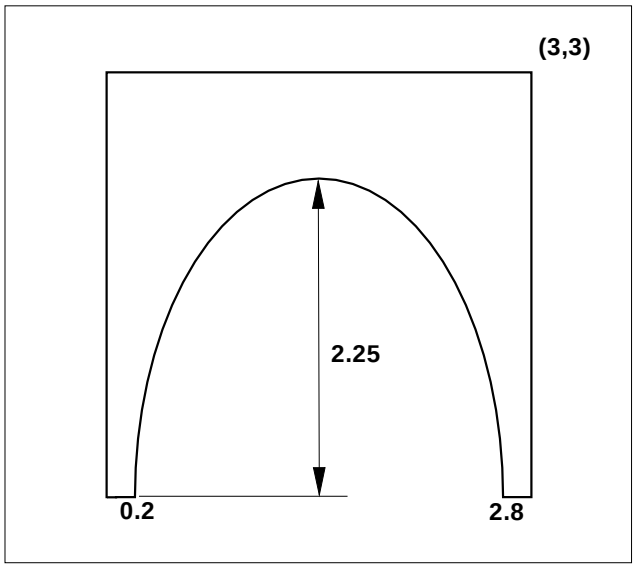
Figs. 4.26(a)-(f) show that all the mapping operators introduced in this chapter generate invalid meshes around the peak of the large spike, except the Winslow operator. Analysing this on the basis of the volume of the physical domains, shows that Fig. 4.24(b) has a smaller volume compared to that of Fig. 4.24(c) which means that the computational mesh in Fig. 4.24(a) will be more squeezed

or deformed to fill Fig. 4.24(b) when compared to Fig. 4.24(c). However, all the methods except the Winslow operator generate inverted cells around the discontinuous point on the spike.

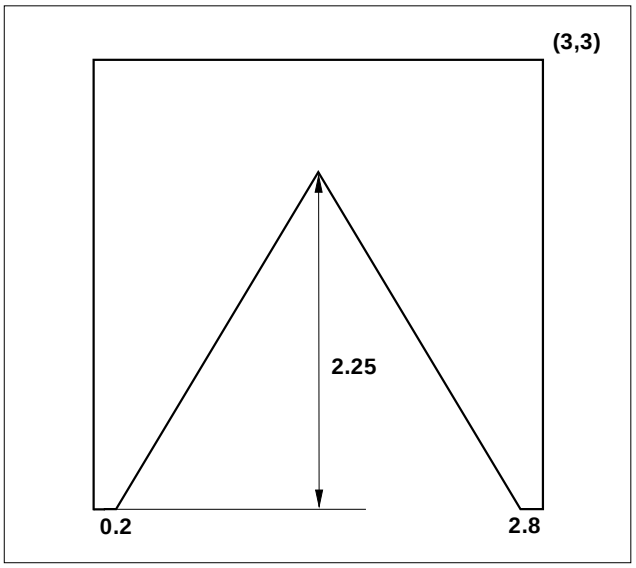
Therefore, the features on the boundary such as discontinuities and variations in curvature, have a direct effect on the resulting mesh in terms of validity, smoothness and quality.



(a)

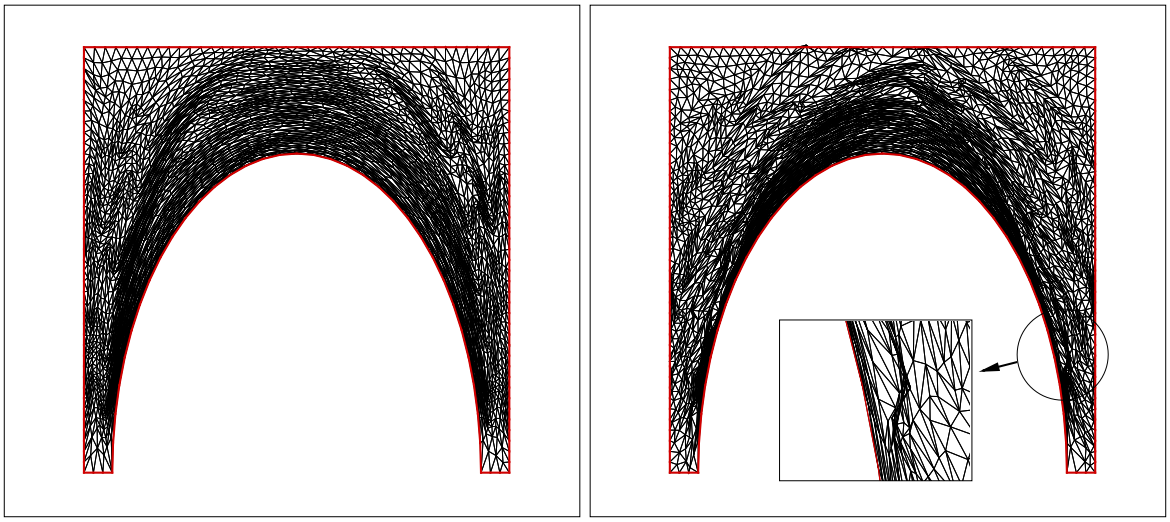


(b)



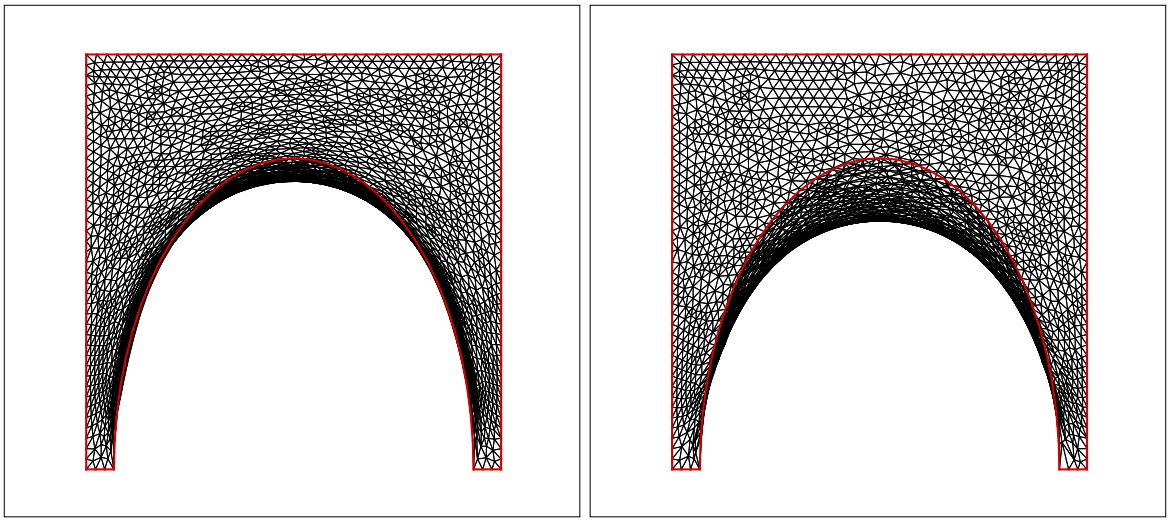
(c)

Figure 4.24 (a) Computational mesh, (b) bump and (c) spike geometries



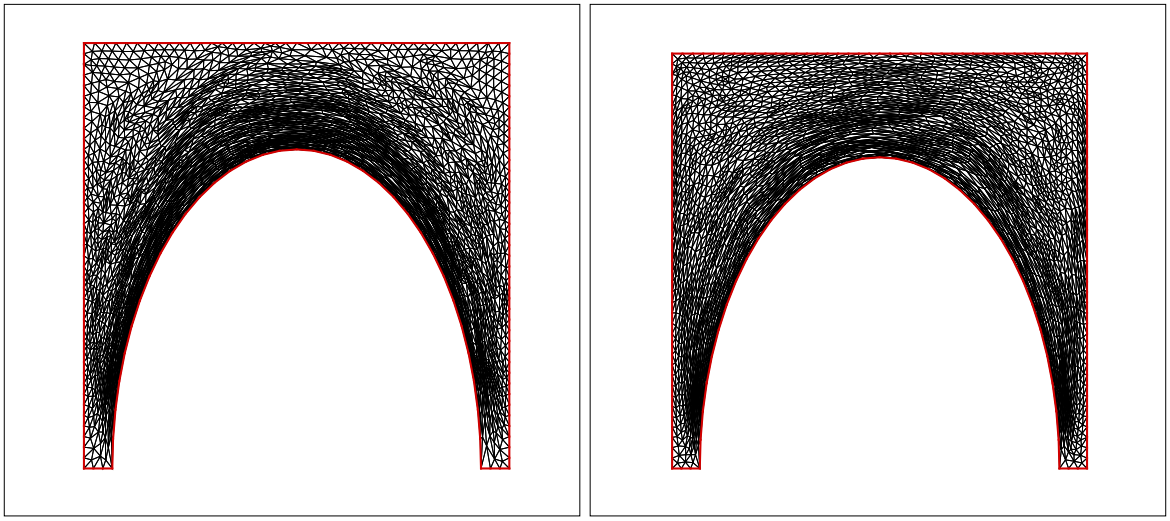
(a) ’A’ functional

(b) ’O’ functional



(c) ’AL’ functional

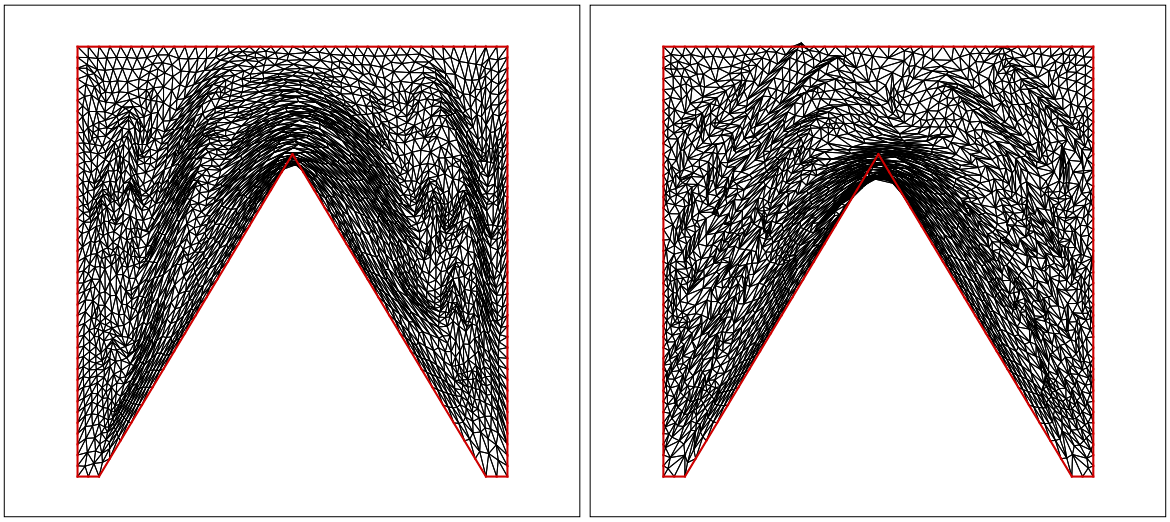
(d) ’OL’ functional



(e) ’AO’ functional

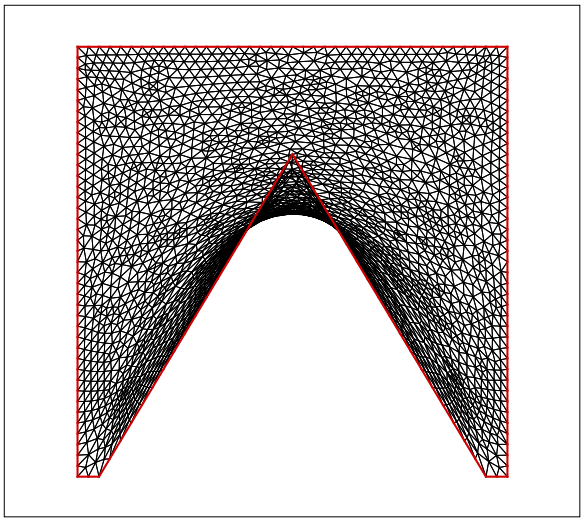
(f) Winslow

Figure 4.25 Comparison of individual and combined functionals around an extreme bump

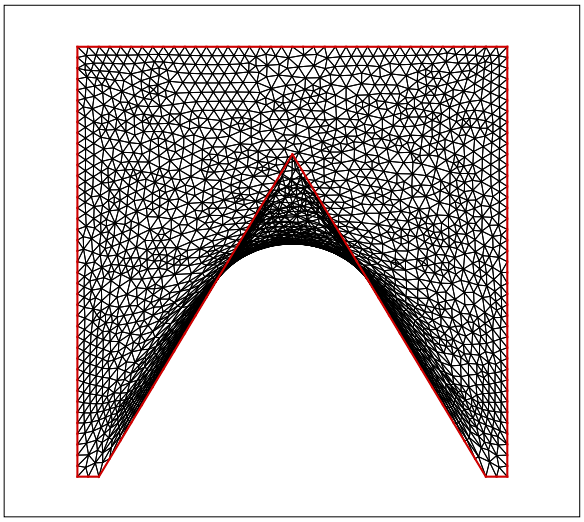


(a) ’A’ functional

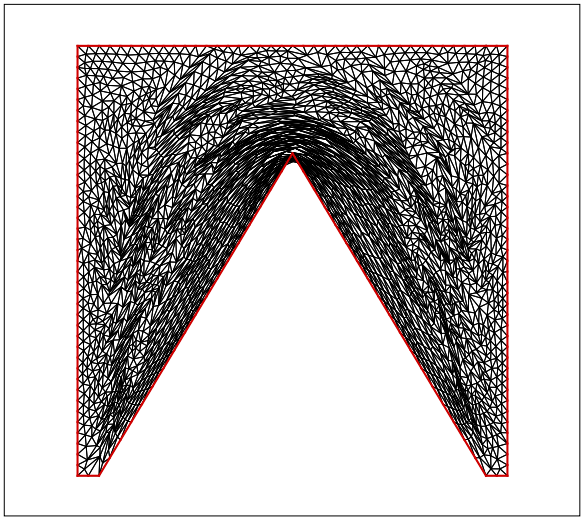
(b) ’O’ functional



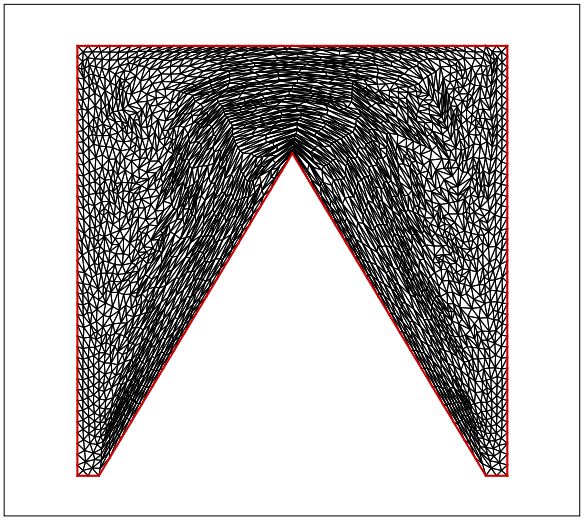
(c) ’AL’ functional



(d) ’OL’ functional



(e) ’AO’ functional



(f) Winslow

Figure 4.26 Comparison of individual and combined functionals around an extreme spike

4.8 Discussion

In the present work, two PDE-based mapping operators were compared, namely the Winslow and the functional-based operators derived from Knupp's work [Knupp & Steinberg (1993)].

According to the resulting mesh obtained from both types of mapping models, the validity, smoothness and quality are directly depend on the defined geometry in the physical space. For instance for four-petal rose geometry, fewer skewed cells appear around the concave corners in domain by AO functional comparing to the Winslow operator, whereas for the spike geometry is the opposite.

On the other hand, the individual functionals of area and orthogonality give nonsmooth mesh because of their type in equations. Also, the length functional (Laplace equation), which is elliptic in type, generates tangled mesh in highly curved regions.

Therefore, according to the cases demonstrated in this chapter, Winslow equation and AO functional will be used in our methodology to handle the mesh motion in physical domain based on sliding cells and mapping domains.

CHAPTER 5

APPLICATION TO MESH MOTION

The methods described in Chapter 2, all share one major characteristic which is that the mesh motion is carried out in physical space. All the methods which preserve the mesh connectivities, finally fail for very large motions because of the fact that the boundary cells remain attached to the boundaries as time evolves. As a remedy, using a slip condition to allow the cells to slide over the boundaries for rigid body motions was described in Chapter 3. However, this method presents difficulties regarding the management of grid valence as required for local element splitting at separation and reattachment points. Another difficulty is the nodal velocity propagation inside the domain to avoid tangled meshes. Consequently, in this chapter, these difficulties are addressed by managing grid motion in computational space, and then, mapping the grid to physical space using a system of PDE operators. Managing cell movement in computational space decreases difficulties in handling physical domain features, especially in regions where boundary curvature varies rapidly. The major goal of this approach is to preserve the mesh connectivity as time evolves, except when the cells split at the leading edge, slide over the boundaries and re-attach at the trailing edge.

This chapter constitute a general framework for large rigid body motions and applied to several examples to illustrate the application of the approach.

5.1 Generic boundary in the computational domain

The unstructured mesh management and mesh sliding technique require a computational mesh with a set of generic boundary definitions that match the element topology of the physical mesh.

An important characteristic is that the computational mesh should be easy to generate and be a valid mesh with positive areas and Jacobians for all cells. The computational mesh should be as generic as possible, and clearly identify the parts of the boundaries that are set in motion. Since boundary movements can be decomposed into a combination of translation and rotation motions, two elementary generic configurations are defined in the computational domain: a slit for translation, and a circle

for rotational motion, shown schematically in Figs. 5.1 and 5.2. More complex trajectories can be obtained as combinations of these.

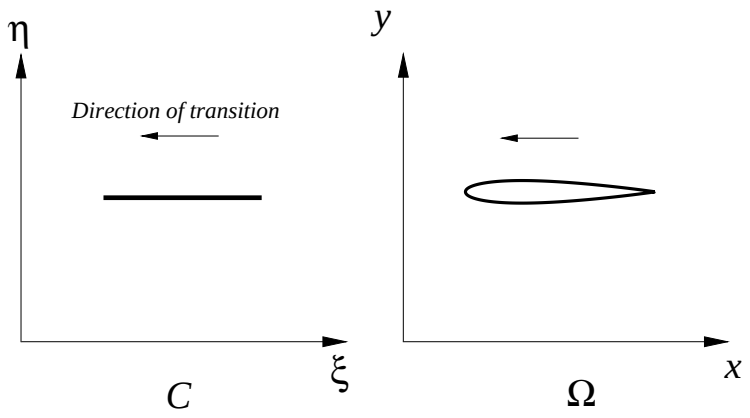


Figure 5.1 Mapping of a generic configuration for translation motion

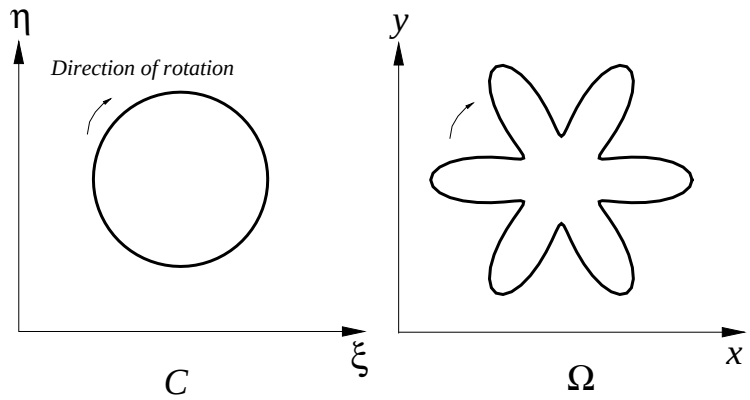


Figure 5.2 Mapping of a generic configuration for rotational motion.

Sliding of the boundaries in computational space $\mathcal{C}$ at each step yields the values of $x(\xi, \eta)$ and $y(\xi, \eta)$ in physical domain Ω by solving the mapping operators presented in Chapter 4. The following sections describe the details of using the generic geometries in computational space for both translation and rotational motions.

5.2 Translational motion

To manage the mesh for translation motion, a computational mesh is created such that the boundary points lie on a slit with zero thickness. These boundary points are displaced or relocated at each step in the computational domain so that they conform to the correct geometry in physical space by applying mapping operators. This relocation is accomplished by updating the topological connections for the cells attached to the boundary. This linear translation is presented in details and applied to the motion of a cylinder in a cavity to illustrate the basic methodology. The effect of large variations in the proximity of boundaries in relative motion is studied using the case of a cylinder moving past an array of bumps.

5.2.1 Sliding a circle inside a cavity

Figure 5.3 shows the link between boundary nodes (circles) and mesh nodes of the body (slit) (squares). The boundary conditions applied to the mesh nodes which have the same coordinates of the corresponding body nodes. In Figs. 5.4 and 5.5, the moving body is shown as a set of circles: single circles at leading and trailing edges and pairs of circles in between. At a given step the motion consists in changing the topological connection of the mesh nodes (squares) to the body nodes (circles) as the body moves through the mesh. For translation motion, the mesh nodes in computational domain remain fixed and the boundary nodes split the cells that lie on the defined trajectory. Each mesh point lying on the moving body has a corresponding node on the boundary, that changes at each time step. For instance, corresponding to mesh node n in computational space is node j on the boundary. In the next step, node $j+1$ becomes the corresponding point for n. The mapping operator will be solved at each time step, but with different values of the boundary conditions. It is necessary to mention that all the dimensions and lengths, except moving boundaries, in both computational and physical domains are equal. This is not a requirement, but is used here to simplify the procedure.

In carrying out this procedure, there is a clear separation of grid motion (topological connection) and domain mapping.

Figures 5.6 illustrates the application of this procedure using the Winslow operator at three time steps. Tracking specific nodes, in both computational and physical spaces (for example nodes 456

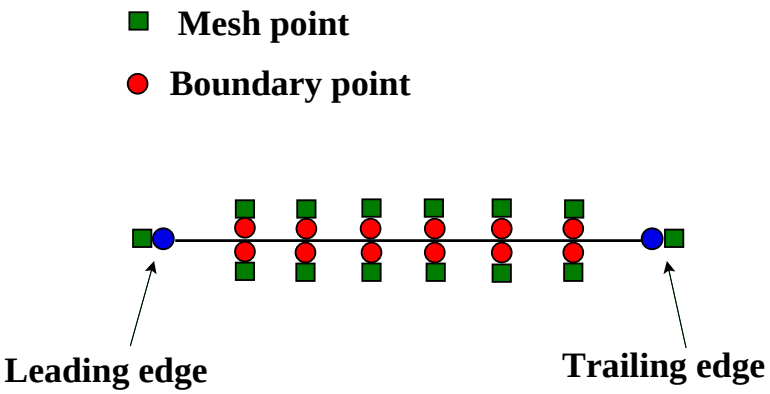


Figure 5.3 Arrangement of the nodes on the slit

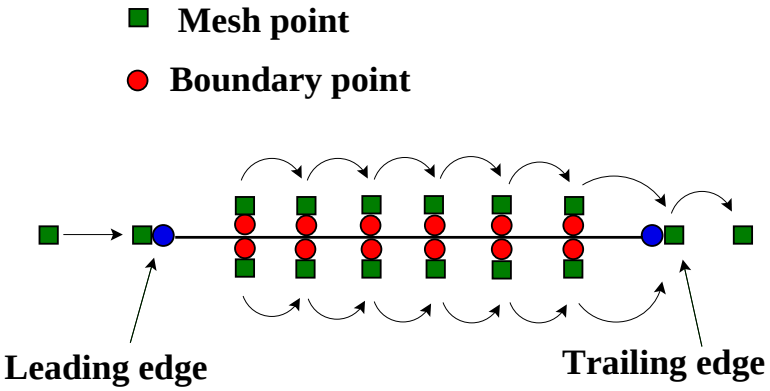


Figure 5.4 Sliding mesh nodes on the slit

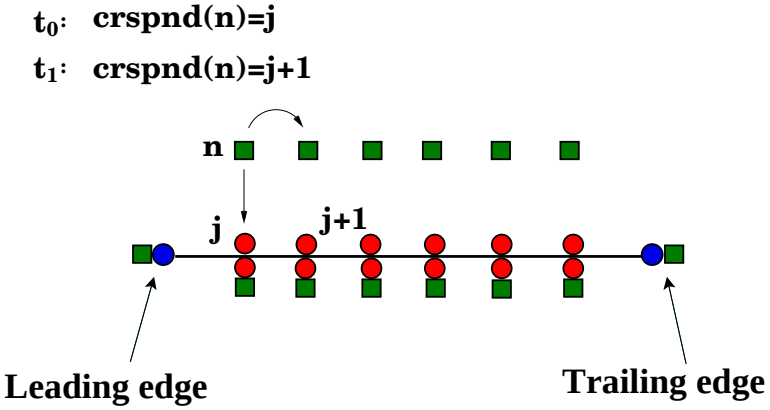


Figure 5.5 Corresponding mesh node on the boundary

and 1366), clearly shows the mesh deformation resulting from the motion of the cylinder.

Comparison of the Winslow operator with the Area, Orthogonality or Area-Orthogonality functionals, is achieved by analyzing the iso-ξ or iso-η contours in (x, y) plane. The smoothness of these results are shown in Figs. 5.7(a) and (b) for the iso-η lines in the computational domain and its corresponding image in the physical space at the intermediate time step of Fig. 5.6(d).

Figures 5.8 - 5.10 show the equivalent resulting meshes and iso-η lines obtained by the Area, Orthogonality and Area-Orthogonality functionals in the physical domain, respectively. As expected from the type of the functional equations and lack of ellipticity, the iso-η lines in these figures are nonsmooth, as are the meshes in the physical domain. Finally, as shown in Fig. 5.11, the Length functional generates an invalid mesh for this test case.

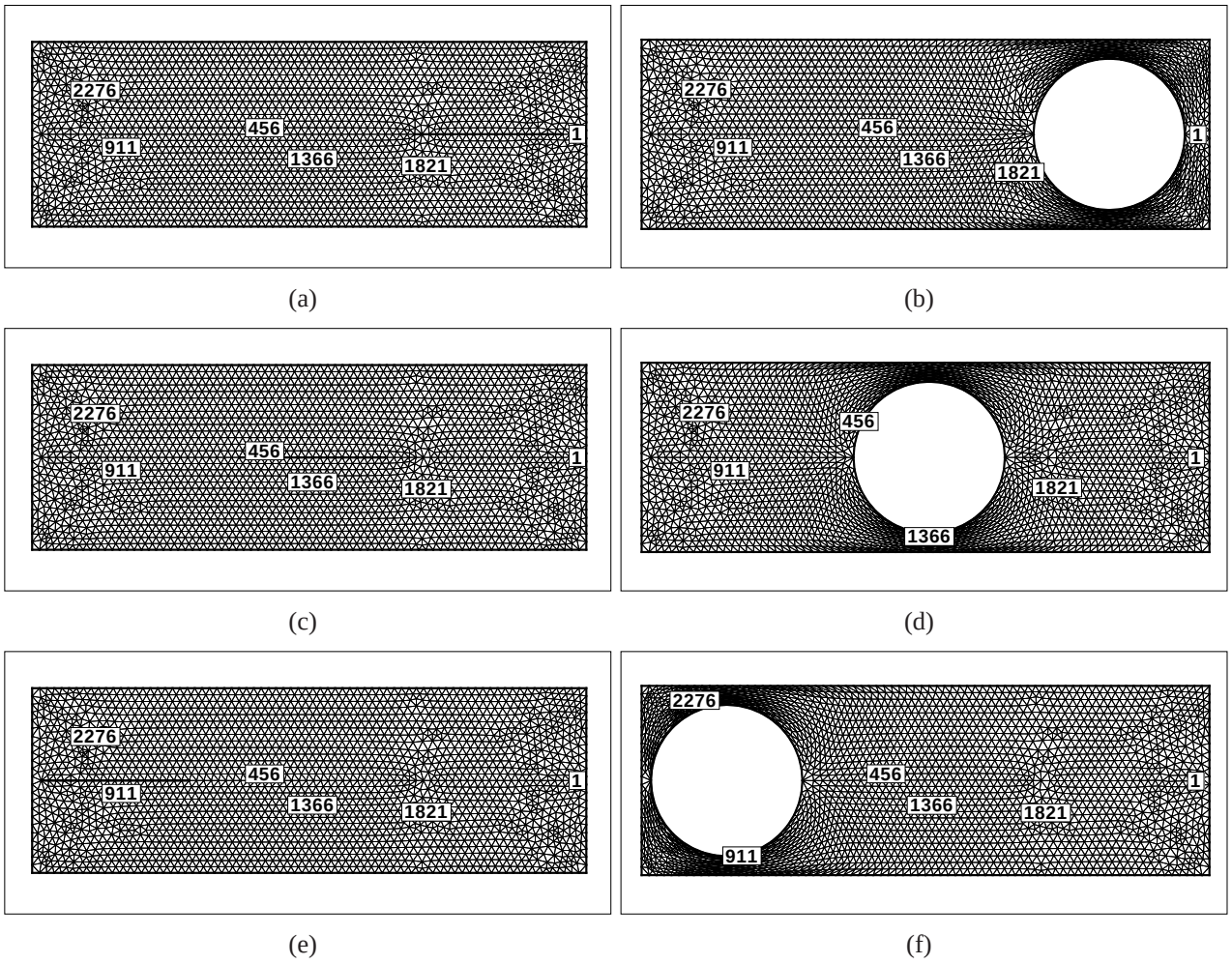
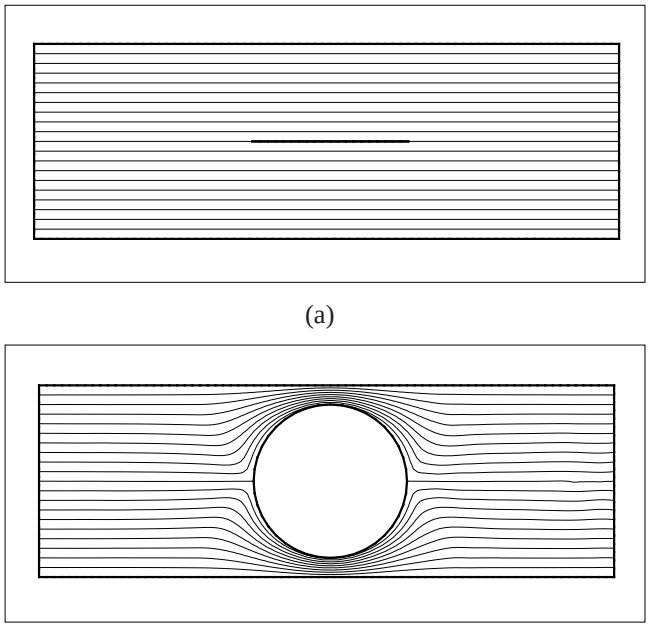


Figure 5.6 Computational and corresponding physical mesh at first, intermediate and final steps



(a)

(b)

Figure 5.7 Iso-η lines in (a) computational and (b) physical domains by Winslow operators

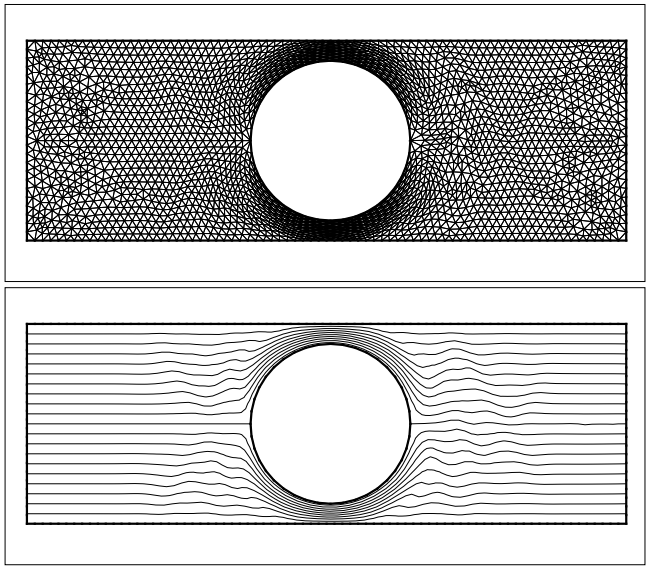


Figure 5.8 Mesh and iso-η lines in physical domain by A functional

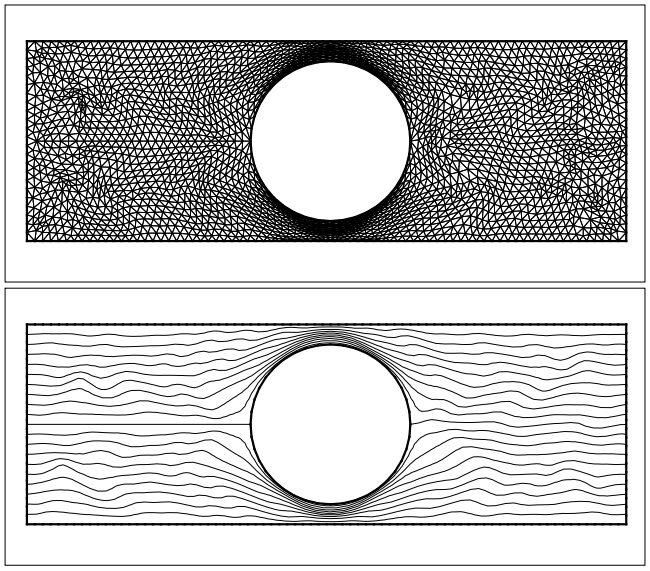


Figure 5.9 Mesh and iso-η lines in physical domain by O functional

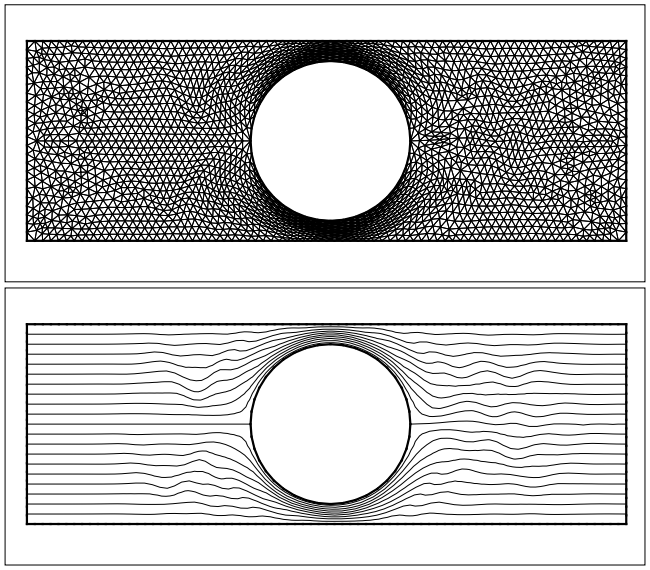


Figure 5.10 Mesh and iso-η lines in physical domain by AO functional

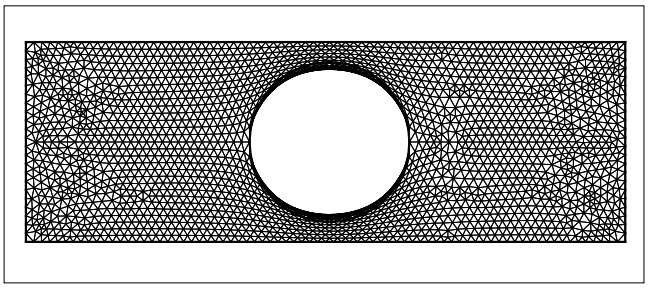
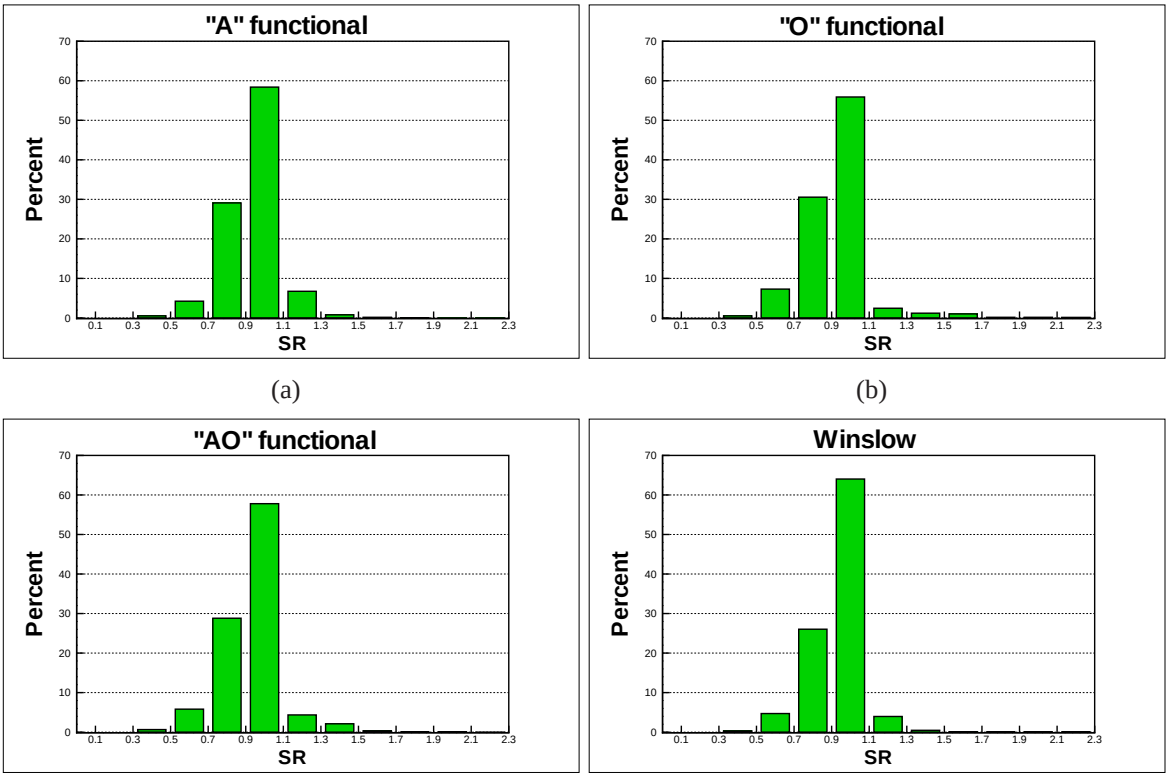


Figure 5.11 Invalid mesh in physical domain by L functional

An analysis of the smoothness of the resulting mesh, based on the different functionals and the Winslow operator, is shown in Figs. 5.12. In this example, the differences between the iso-η contours are clearly reflected in SR values as shown in Figs. 5.12. Finally, global smoothness can be assessed from the Smoothness Factors given in Table 5.1.



(a) (b)

(c) (d)

Figure 5.12 Comparison of mesh smoothness around the moving circle at the intermediate step for different mapping operators

It can be observed that the Winslow operator has the maximum value for the SF, while the Orthog-

onality functional has the lowest value. Interestingly, in this specific example, the Area functional gives a smoother grid than the Area-Orthogonality functional.

Fig. 5.13 shows the mesh quality at the intermediate step based on the different mapping operators. Although the Winslow operator generates the smoothest mesh, the Area and Area-Orthogonality functionals generate better quality meshes based on the minimum angle criteria, while the Orthogonality functional gives the poorest one. Comparison of the SF values and mesh quality, indicates that improving one criteria will not necessarily improve the other. Also, the resulting mesh is very dependent on the characteristics of the physical boundaries, in addition to the mapping operators.

5.2.2 Sliding a circle past obstacles

The effect of proximity of boundaries in large relative motion is illustrated by the motion of an object moving past an array of bumps in a two-dimensional mesh. The meshes in both computational and physical spaces are shown in Fig. 5.14-5.18 where the physical mesh is obtained by using the Winslow operator. The minimum distance between the moving circle and the peak of the bumps is $0.2R_C$ where R_C is the circle's radius. The zoomed views of the moving circle show how the isotropic cells between the slit in computational domain and the bumps are flattened in order to preserve their validity. Figure 5.19 shows a detailed view of the mesh in physical space when the circle passes over a bump and moves toward the following one, in four different motion steps.

Table 5.1 Values of SF for the sliding circle at intermediate step, case in Fig. 5.6

Operator	Smoothness Factor
Winslow Fig. 5.6 (d)	0.906
A Fig. 5.8	0.868
O Fig. 5.9	0.734
AO Fig. 5.10	0.851

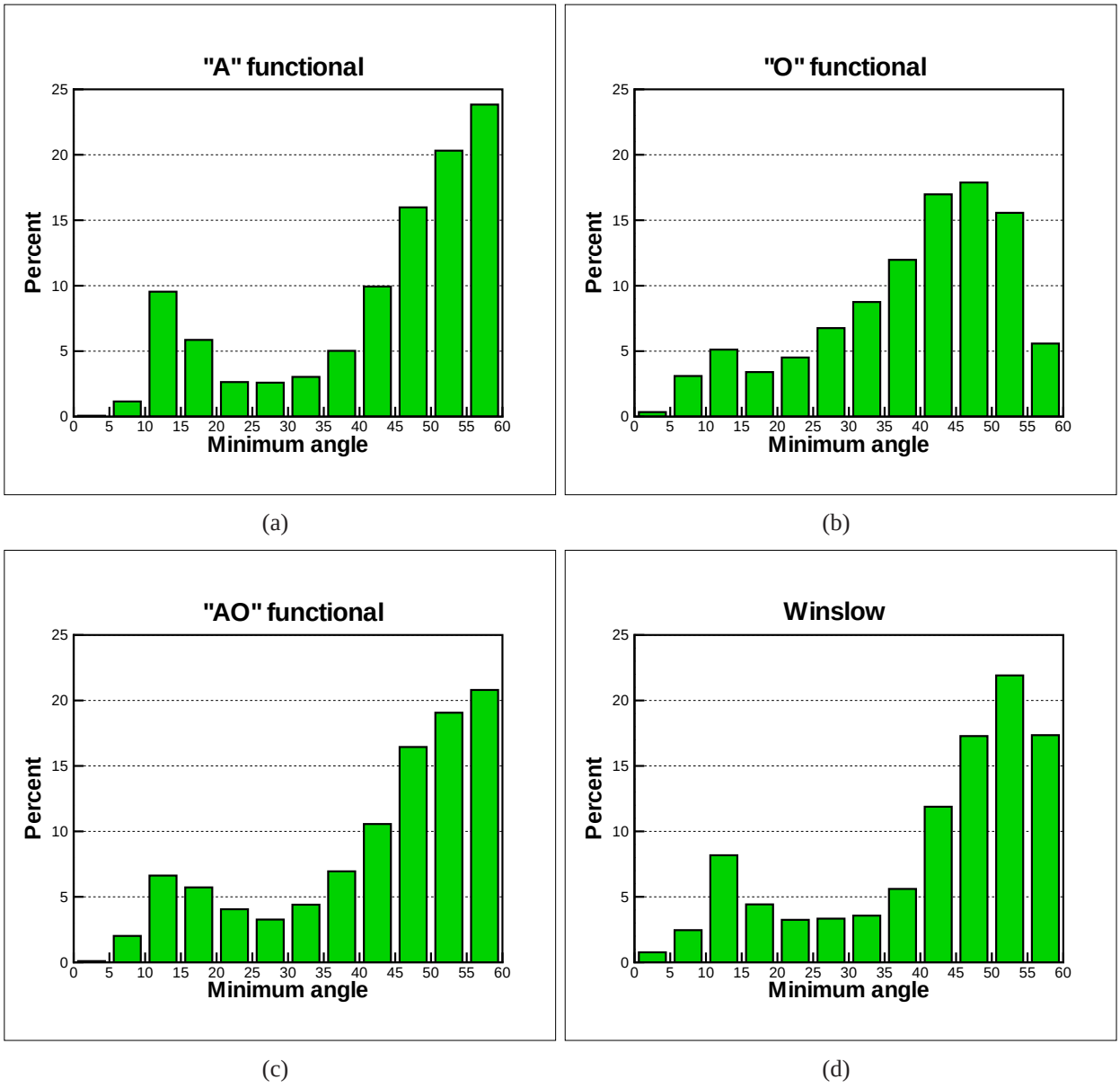
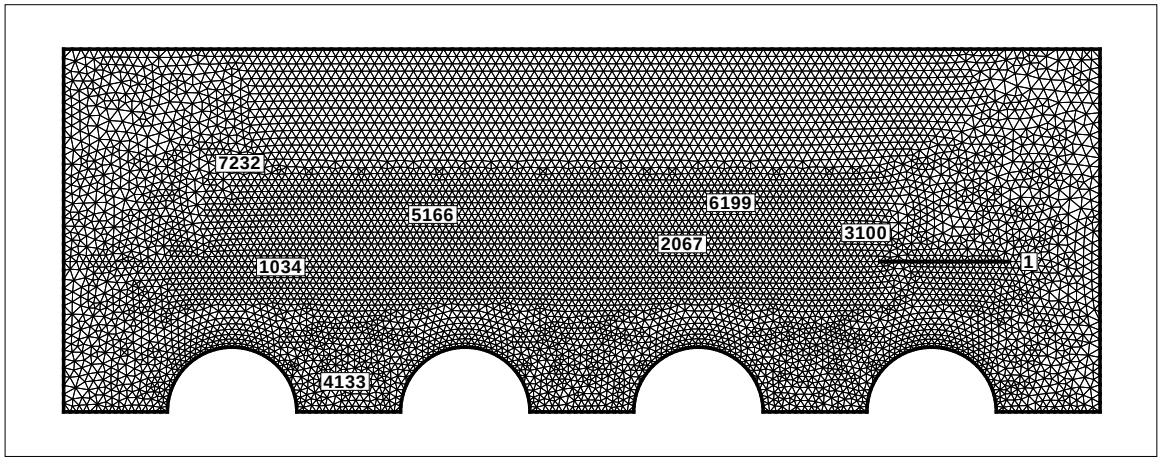
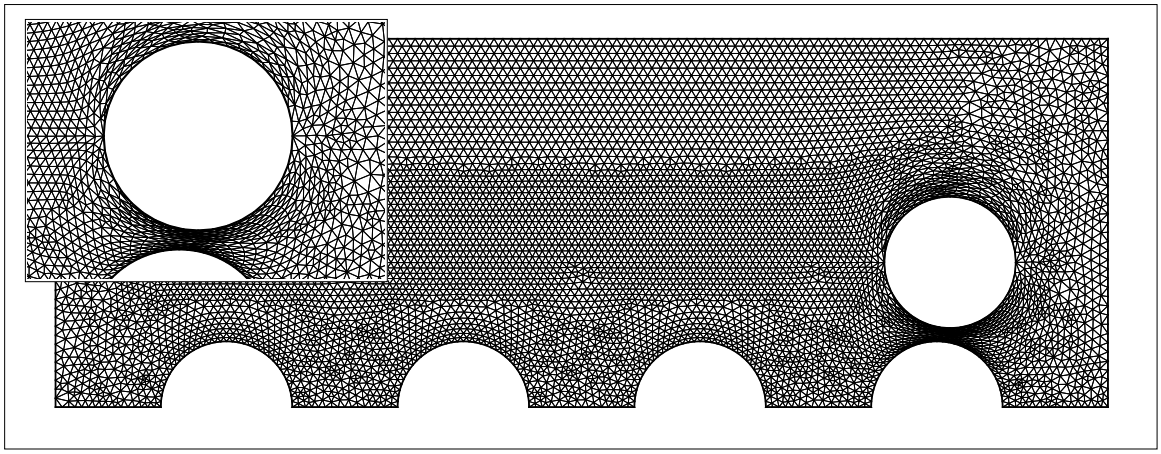


Figure 5.13 Comparison of mesh quality around the moving circle at the intermediate step for different mapping operators



(a)



(b)

Figure 5.14 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=1

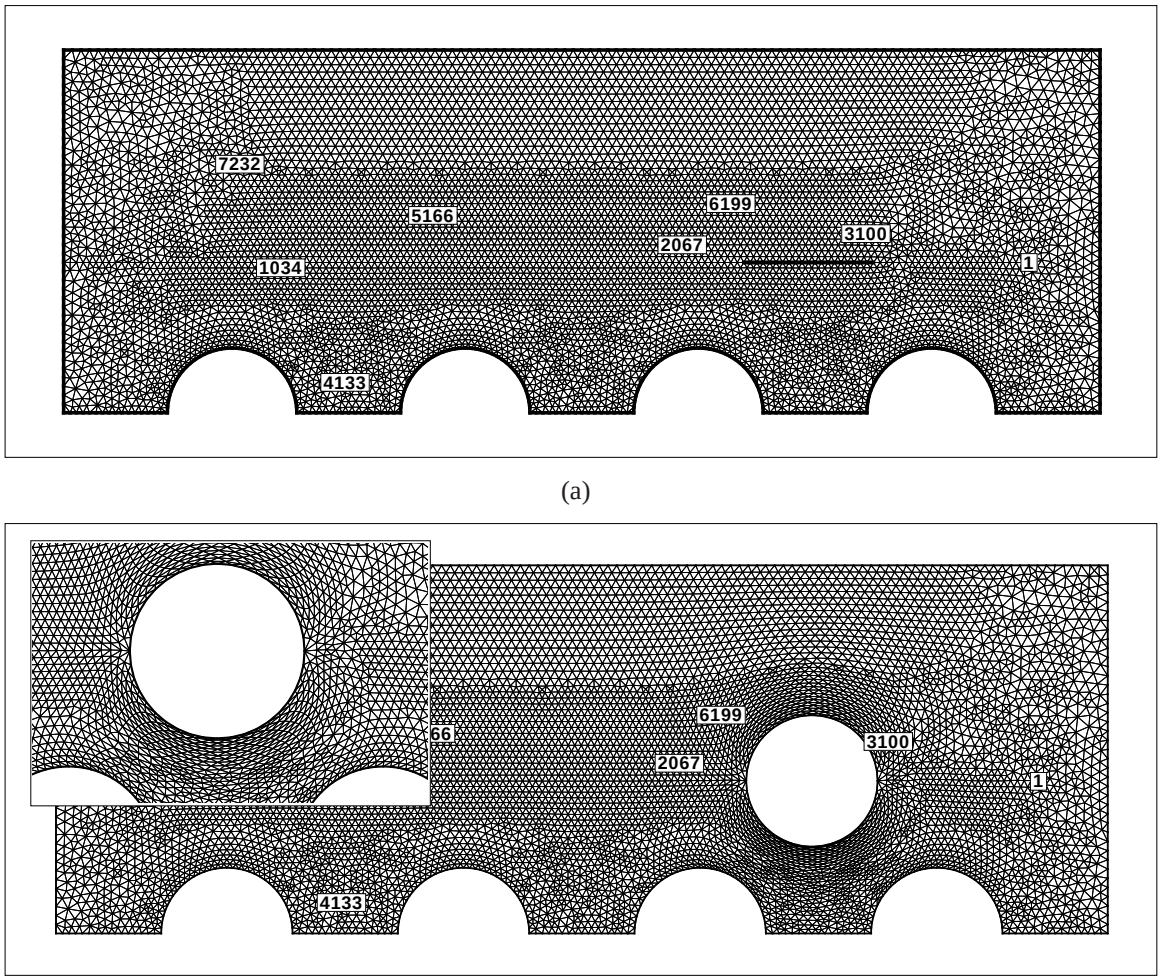
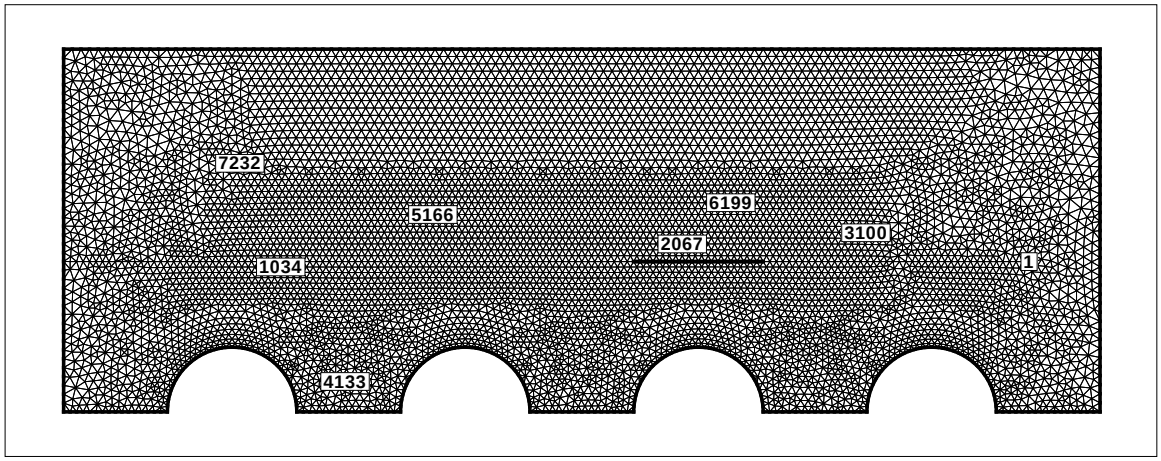
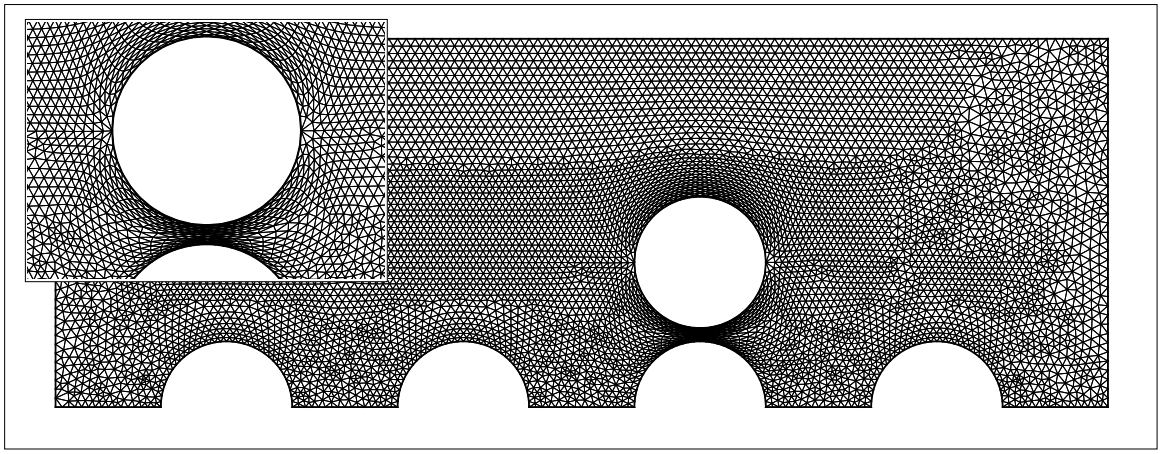


Figure 5.15 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=25

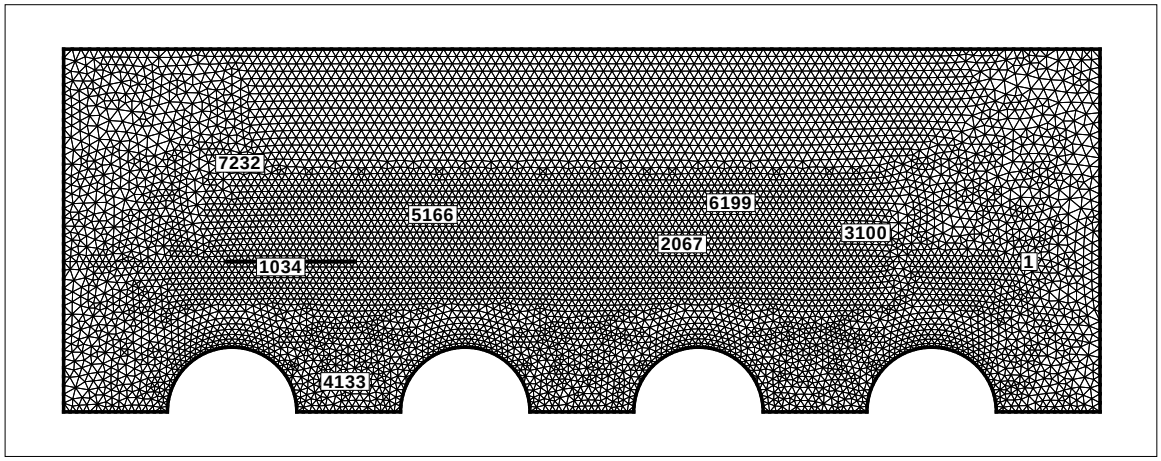


(a)

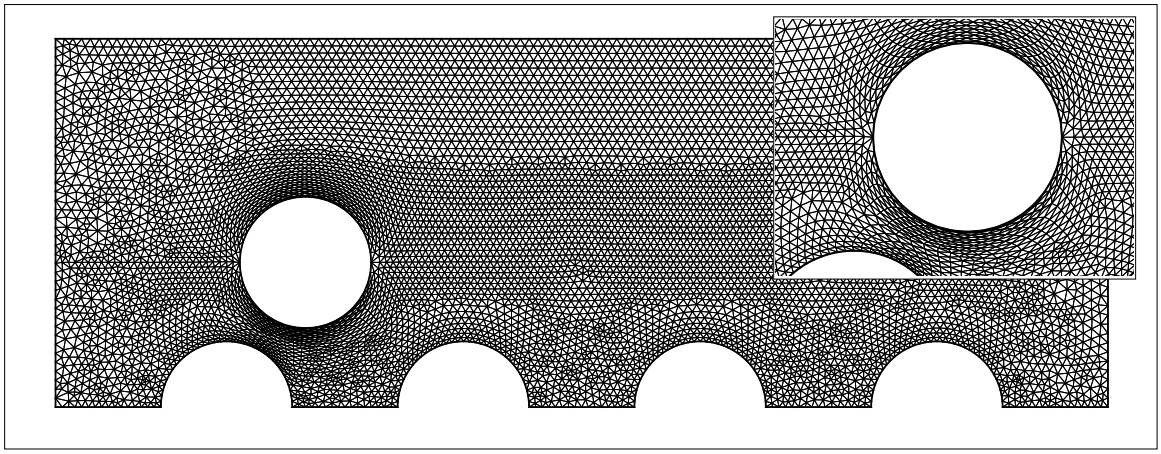


(b)

Figure 5.16 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=42

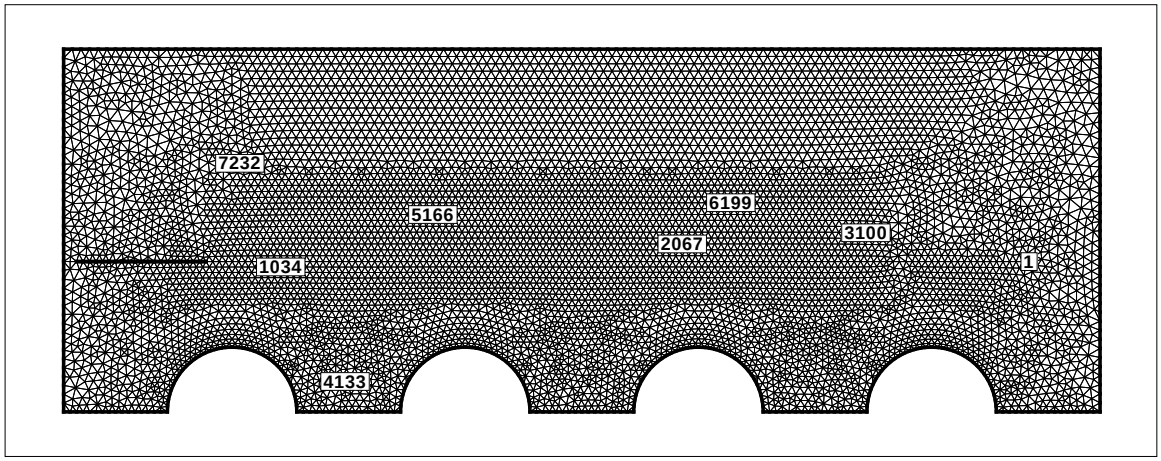


(a)

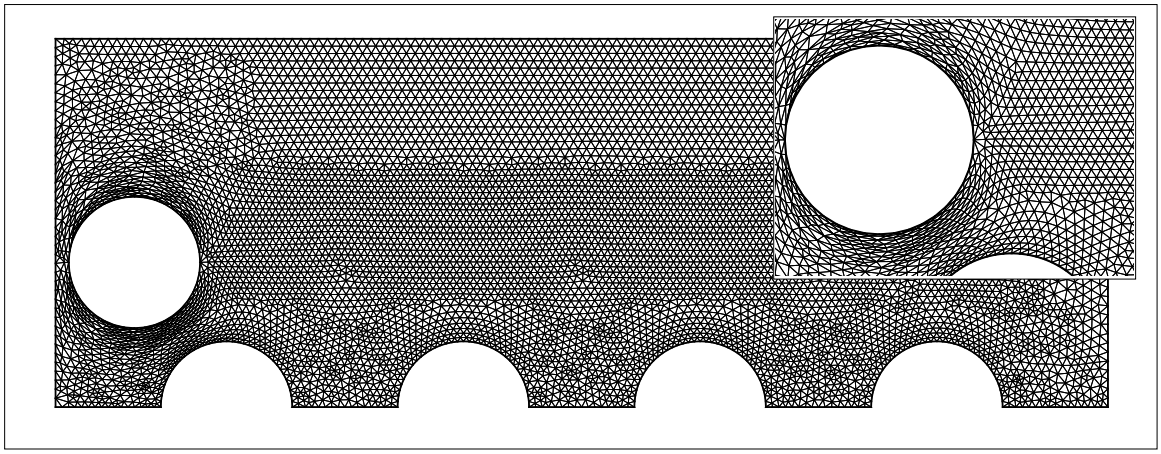


(b)

Figure 5.17 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=105

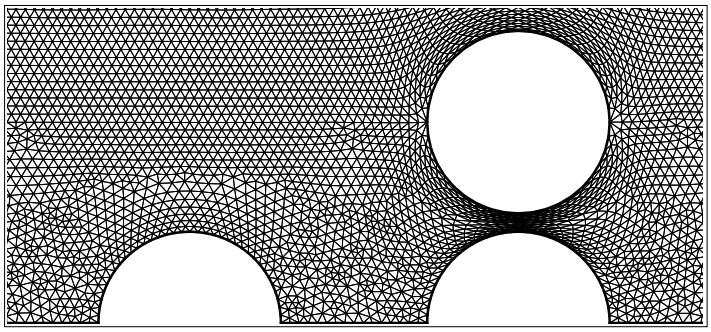


(a)

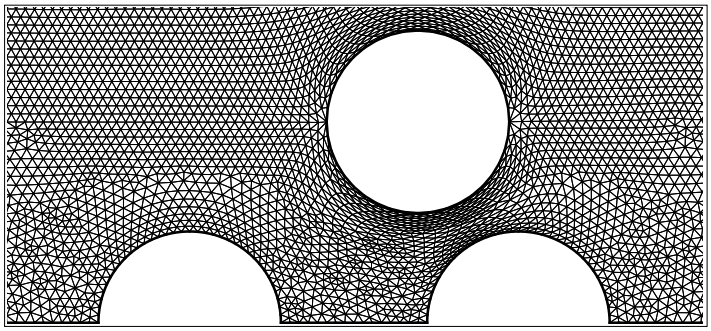


(b)

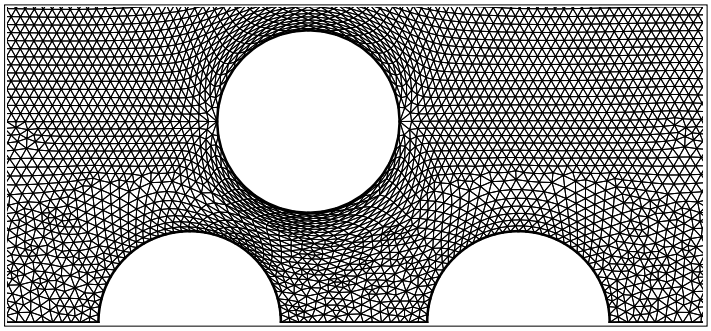
Figure 5.18 Modelling the boundary proximity using the Winslow operator (a) computational and (b) physical domain, t=128



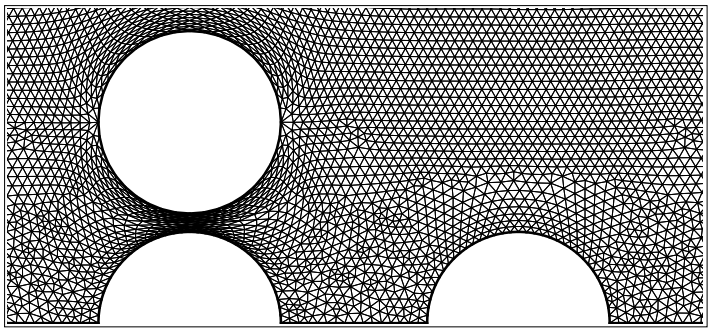
(a)



(b)



(c)



(d)

Figure 5.19 Modelling the boundary proximity using the Winslow operator at four different time steps

5.3 Rotary motion

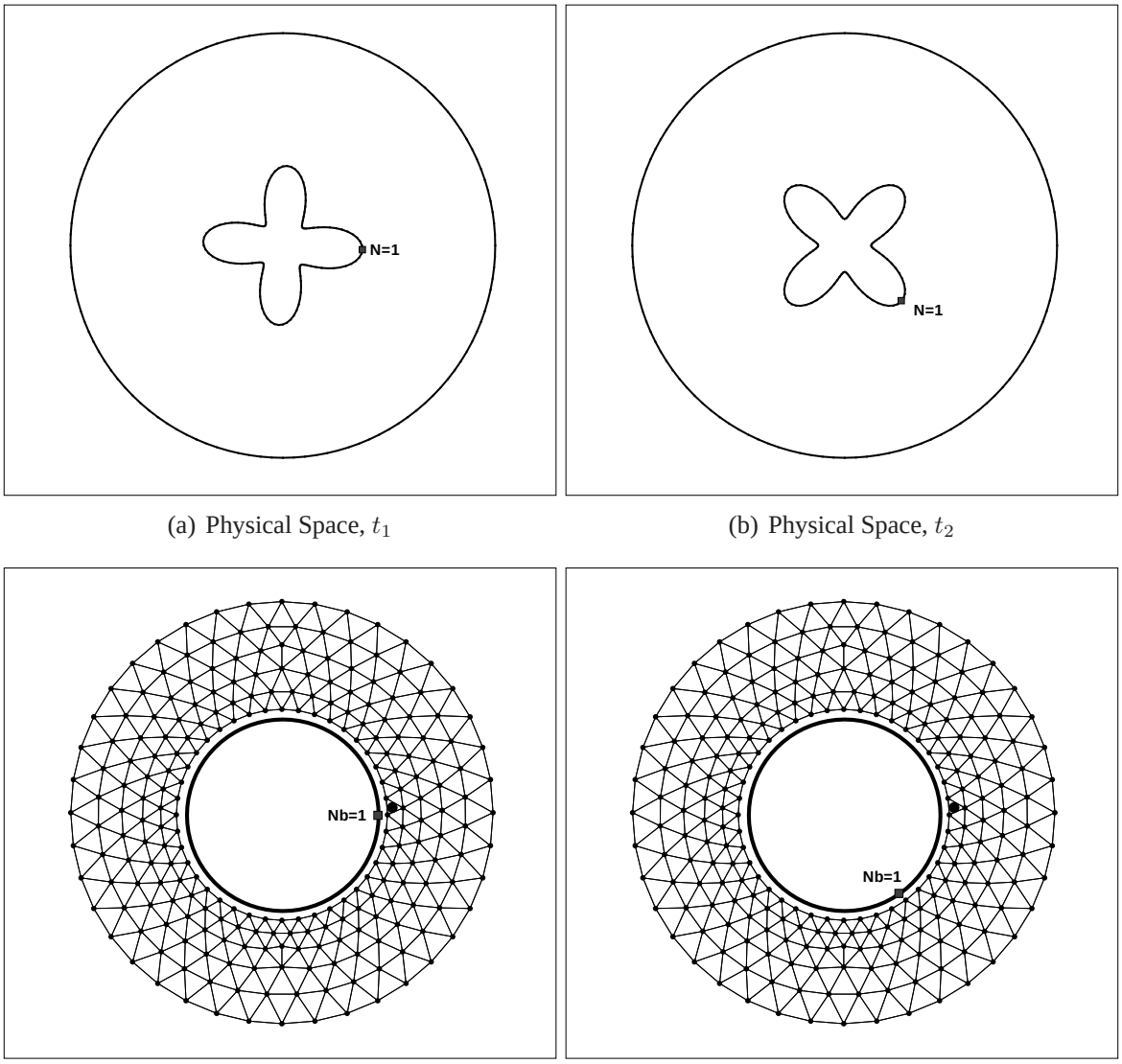
Rotational motion is obtained by rotating about the circle's center which differs from the translation motion. This rotary motion is realized by the generic configuration of a circle rotating inside a mesh in computational space, (ξ, η), as shown in Fig. 5.20. The topological connections between the mesh nodes and the body nodes are updated at 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, thus maintaining the same connectivity. Also, to have an equivalent motion or Δt in both $\mathcal{C}$ and Ω spaces, the angular rotation can be controlled in $\mathcal{C}$ with respect to the motion defined in Ω. 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 $\mathcal{C}$-space at each iteration with the evolving boundary conditions in Ω. This procedure is repeated until the trajectory is completed in physical space. This is illustrated in Fig. 5.20 by the movement of the boundary nodes $Nb, N = 1$ in $\mathcal{C}$ and Ω respectively.

This process of sliding boundary nodes is illustrated in more details in Fig. 5.21, which shows for element number 350 (at the top of the diagram) how the boundary pointers change from $61 - 62$ to $60 - 61$, and finally to $59 - 60$ in the three consecutive steps in both the computational space and the mapped physical space.

To demonstrate the applicability of the proposed methodology, the rotation of three complex shapes with highly curved regions is presented in the following sections.

5.3.1 Rotation of a NACA0012 airfoil

The first example is the rotation of an airfoil shown in Fig. 5.22. The results obtained by the Winslow operators are shown at six angular positions. The mesh motion can be understood by following the path of node 1 which represents a boundary node on the airfoil and cell number 530 which slides on the boundary. In contrast to the RBF method presented in Chapter 2, the whole mesh maintains its isotropy for this severe rotation.



(a) Physical Space, t_1 (b) Physical Space, t_2

(c) Computational Space, t_1 (d) Computational Space, t_2

Figure 5.20 Rotating physical boundary and sliding computational boundary over the mesh points

5.3.2 Four-petal rose rotation

A more complex mapping is illustrated in Fig. 5.23, showing the mesh around the generic circle in computational space, mapped to physical space around a four-petal rose. The mesh motion methodology can be visualized by following the movement of cell 1 and node 1 through three successive time steps. As it can be seen from the figures, when the circle slides in the (ξ, η)-plane, the one-to-one mapping procedure maintains the cell connections in (x, y) plane and only changes the node coordinates. By solving Eqns. 4.7, the cell shapes and their validity are controlled at each iteration. The

Euler-Lagrange equation for the AO functional is used for this example.

5.3.3 Relative rotary motion

The third example is the rotation of two four-petal roses in opposite directions as shown schematically in Fig. 5.24. As it can be seen from this figure, the moving boundaries are two circles in computational space, mapped to two four-petal roses in physical space. For this case, the circles are discretized with the same number of nodes.

The objective for this case is to illustrate relative motion with a wide range of amplitudes for the proximity of boundaries, while maintaining a fixed connectivity. This complex motion is shown in Fig. 5.25 at three different steps. These results were obtained using the Winslow equations. As is shown in these figures, the cells in the computational domain at different time steps are fixed except for the pointers on the boundaries. For example, tracking boundary nodes 1 and 240 in computational space at different time steps shows how these two circles are sliding inside the computational mesh. In addition, cell number 1, as a boundary cell number in the physical space, reveals how the boundary cells slide over the surfaces.

Figure 5.26 shows the evolution of the SF values for a complete rotation. As it can be seen by the figure, the SF values are consistent in the sense that they are respectively repeated at each 90^o whereas the general variation for 360^o rotation is bounded between 0.852 and 0.856.

5.4 Combination of translational and rotational motion

All individual methods presented in this thesis have limitations, but combination of these methods, somehow, may cover their limitations. For instance, the major constraint of the RBF or linear elasticity methods, as explained in Chapter 2, is attaching the cells to the boundaries as motion evolves. Therefore, generating skewed cells in domain is inevitable. This constraint can be released by using sliding cells as a preliminary mesh motion technique to handle wider applications such as moving boundaries on a curvilinear trajectories.

Generally, an arbitrary motion of the boundaries in the physical space can be decomposed into two

elementary motions, translation and rotation. These two types of motions have been explored and studied extensively and a framework for grid motion is prepared in this study.

In the presented method, combination of these two elementary motions becomes possible by defining a parametric space, (u,v), between the computatioal and physical spaces, as shown in Fig. 5.27. The first step is sliding the computational mesh on the generic boundary. This is followed by mapping the computatioal to the parametric mesh which the location of the boundary in parametric space is corresponding to the location of the generic boundary defined in the computatioal domain. Finally, the parametric mesh is deformed using a secondary mesh motion technique, RBF, to replace the boundary on its curvilinear trajectory. Two complex trajectories for a NACA0012 airfoil are presented below to demonstrate the potential of this combination technique for handling large body motions.

5.4.1 Sliding a NACA0012 airfoil on a sinusoidal trajectory

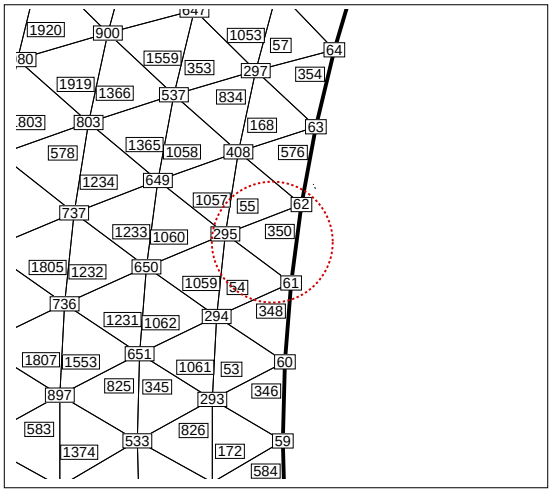
This novel approach has been applied for a NACA0012 airfoil with an oscillatory motion and large amplitude in the physical domain. The trajectory is shown in Fig. 5.28 which is a sinusoidal motion in two sequential periods. The sliding direction of the slit and airfoil in the computational and parametric spaces, are shown in Fig. 5.29. Figure 5.30 shows the parametric mesh and sliding motion in six different steps. This mesh is transfered from computational space using the Winslow operator at each motion step. Figure 5.31 reveals the corresponding deformed mesh in the physical space by applying the RBF technique on the parametric mesh.

To evaluate the resulting mesh, the SF values are calculated for the entire mesh at each step and shown in Fig. 5.32. As it can be seen by the figure, the Smoothness Factor varies by less than 0.7% while its amounts remain always above 0.9. This indicates the strength of the mapping operators which maps the slit to an airfoil in the parametric space using the Winslow operator, and also reveals how RBF technique maintains the grid smoothness for highly deformed meshes in the physical space.

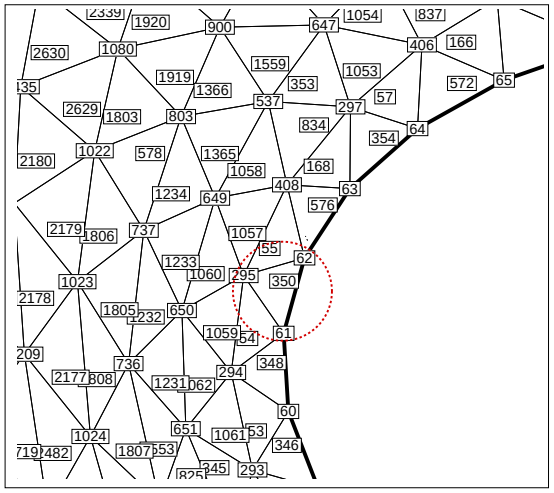
5.4.2 Cobra-like motion of a NACA0012 airfoil

A Cobra-like motion is used to demonstrate sequentially use of the sliding and mesh deformation techniques. In Fig. 5.33, this prescribed motion is shown where the incidence angle of the airfoil

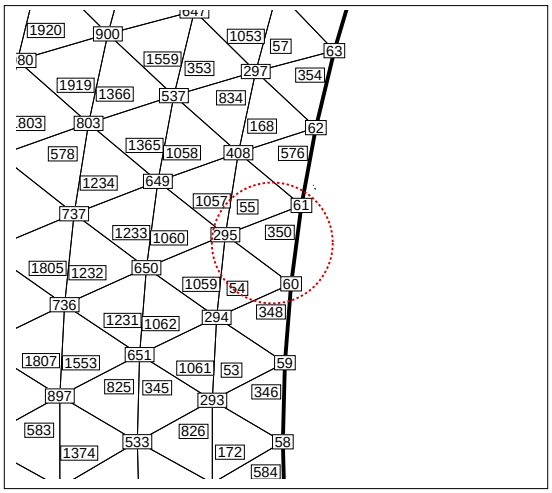
increases gradually to the vertical position and dropping back to the horizontal situation. Again in this example, the linear motion of the slit in negative x-direction takes place in the computational domain and mapped to the parametric space using the Winslow equation as shown in Fig. 5.30. This is followed by deforming the parametric mesh, according to the defined trajectory in Fig. 5.33, to the physical space using the RBF technique. This type of motion illustrates that the combination of both sliding and RBF techniques is a plausible method to perform this type of complex motion by maintaining the cells connectivities except where the boundary splits the mesh. The resulting mesh in physical space is shown in Fig. 5.34 in six different moving steps. The change of the smoothness factor, shown in Fig. 5.35, is about 0.76% and varies between 0.909 and 0.9014 which shows the mesh is indeed smooth during the motion.



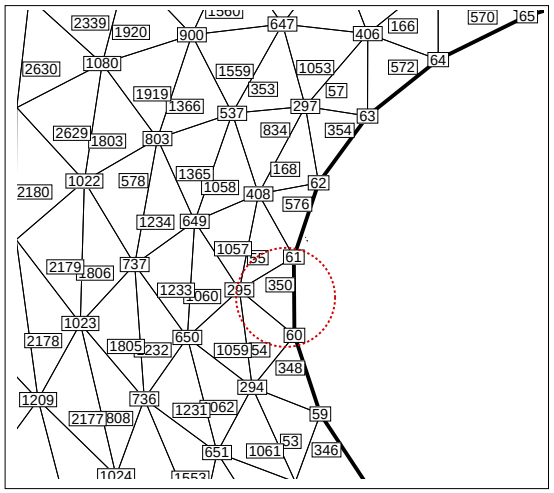
(a) Computational Space, t_1



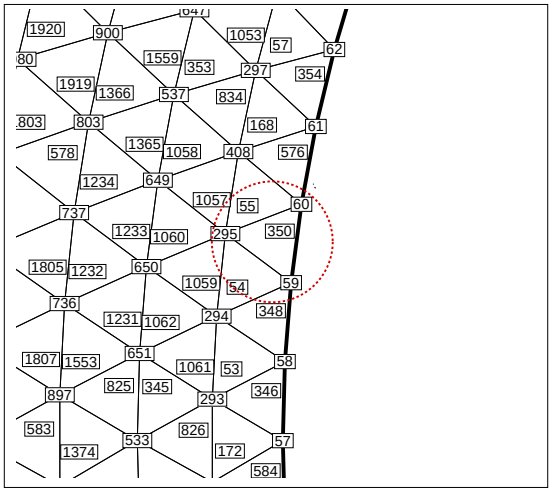
(b) Physical space, t_1



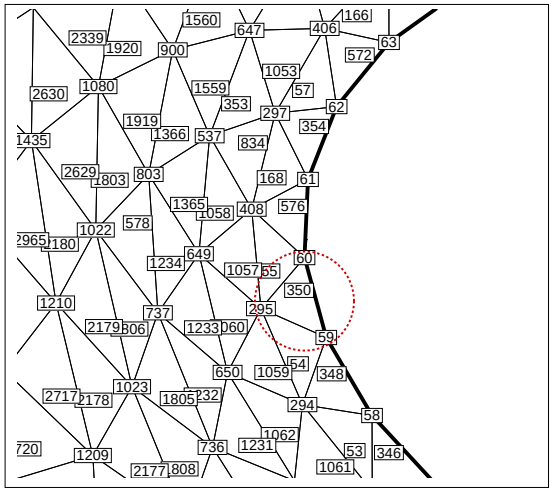
(c) Computational Space, t_2



(d) Physical space, t_2

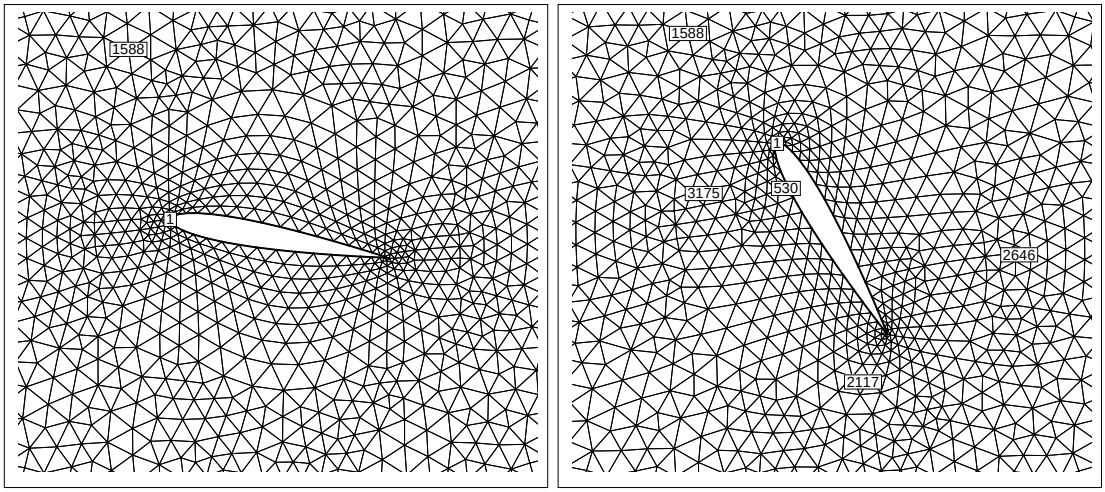


(e) Computational Space, t_3

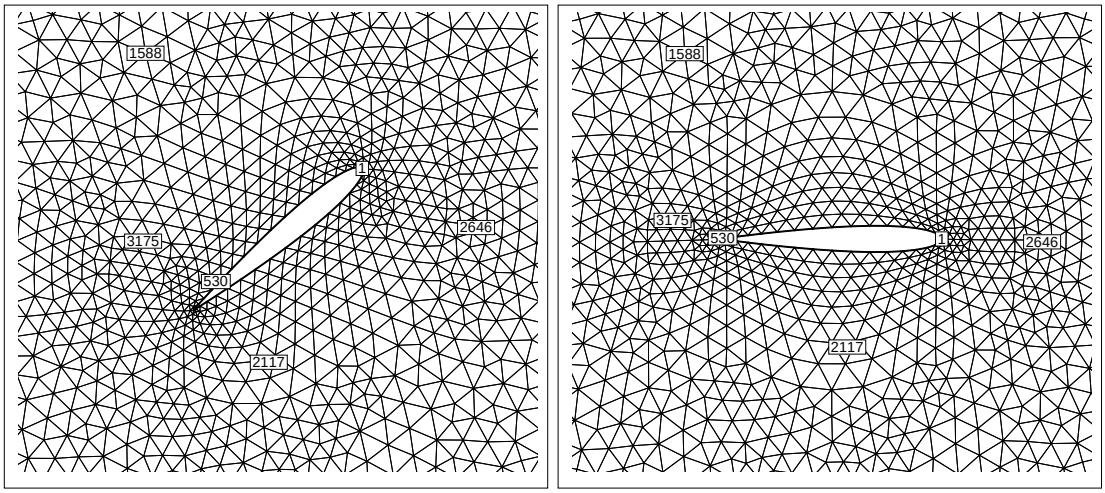


(f) Physical space, t_3

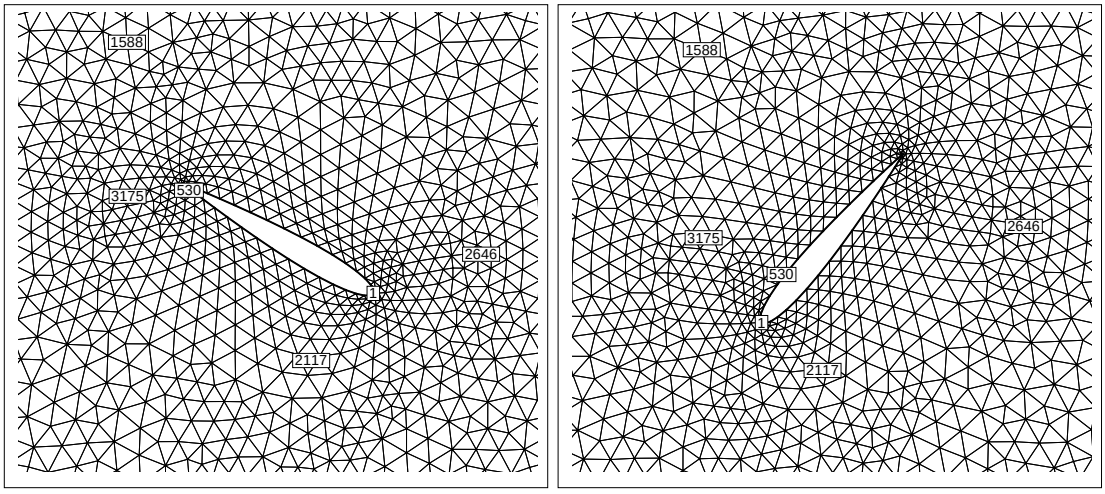
Figure 5.21 Updating the node numbering of the cells attached to the boundary as the result of sliding the boundary for three consecutive time steps



(a) t_1 (b) t_7

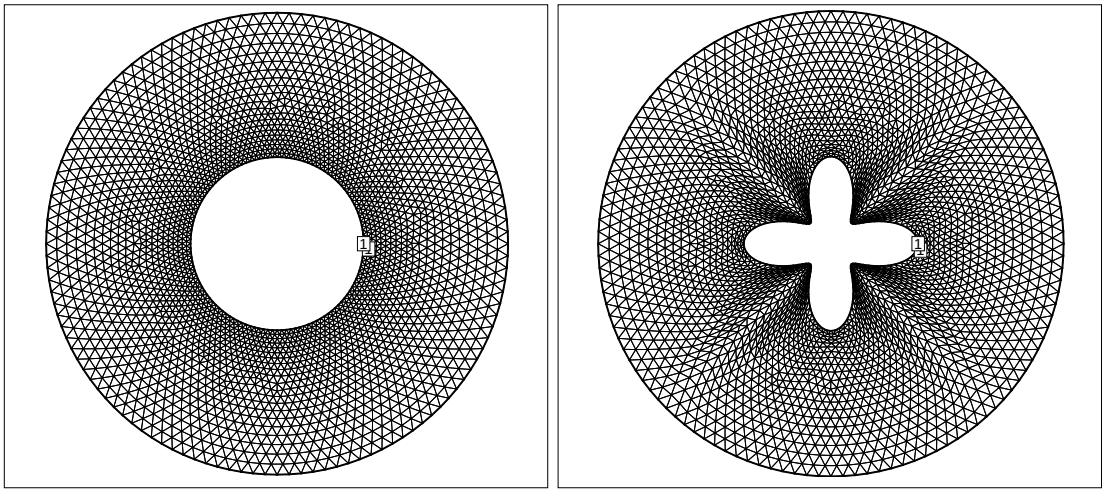


(c) t_{12} (d) t_{19}

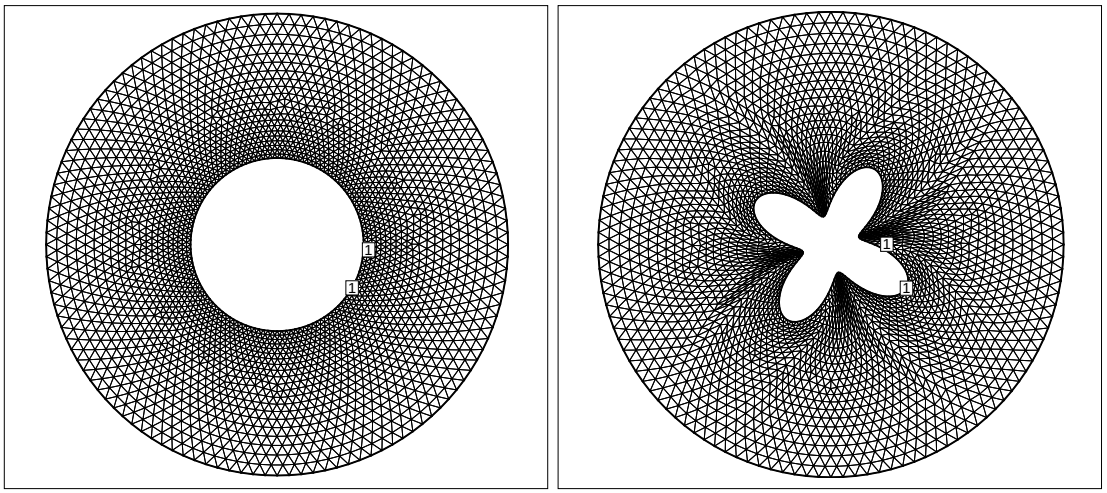


(e) t_{22} (f) t_{32}

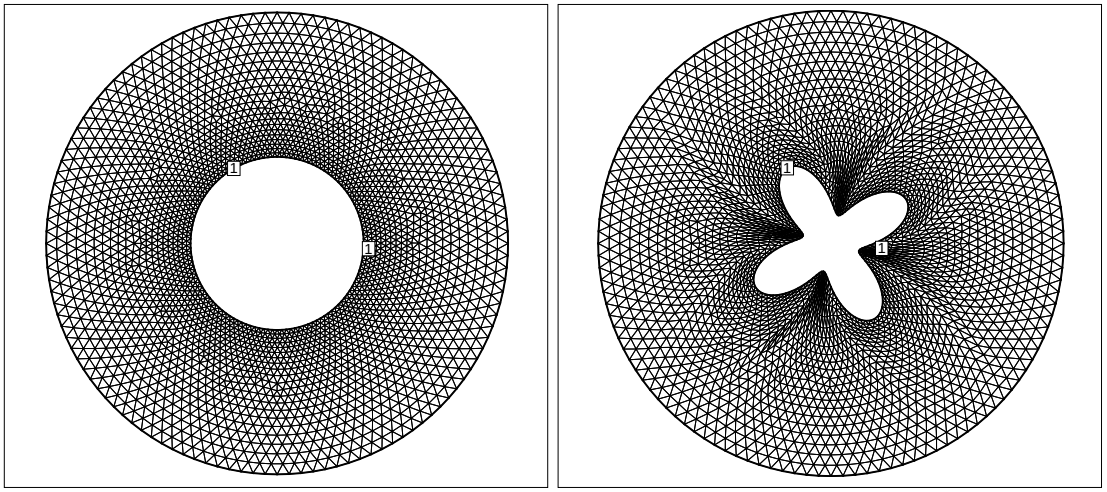
Figure 5.22 Rotating and sliding the cells around an airfoil in the physical domain for six different positions



(a) Computational Space, t_1 (b) Physical space, t_1

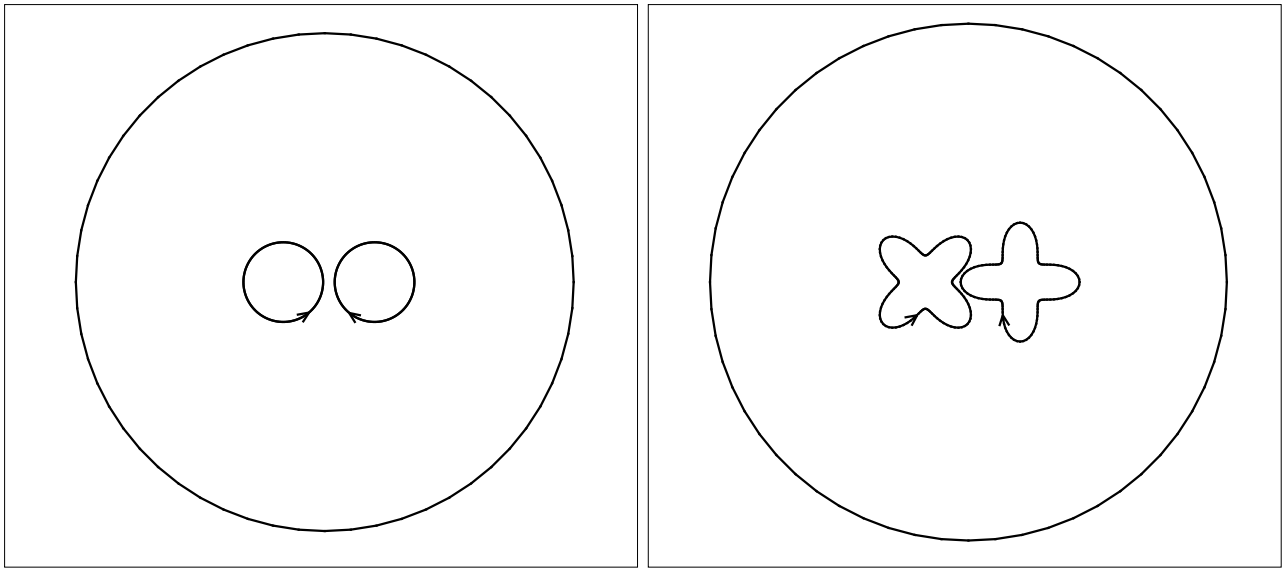


(c) Computational Space, t_{10} (d) Physical space, t_{10}



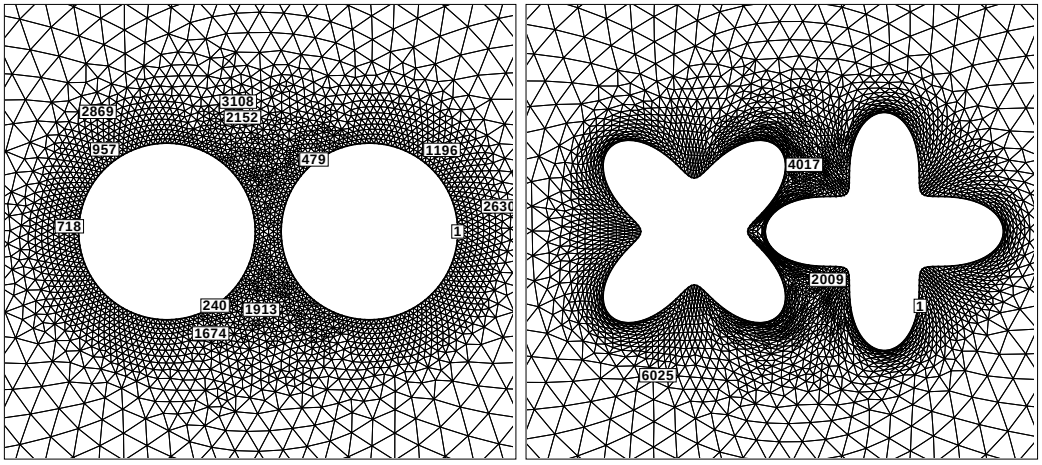
(e) Computational Space, t_{80} (f) Physical space, t_{80}

Figure 5.23 Sliding the cells around the boundary in computational and physical domain at three different steps

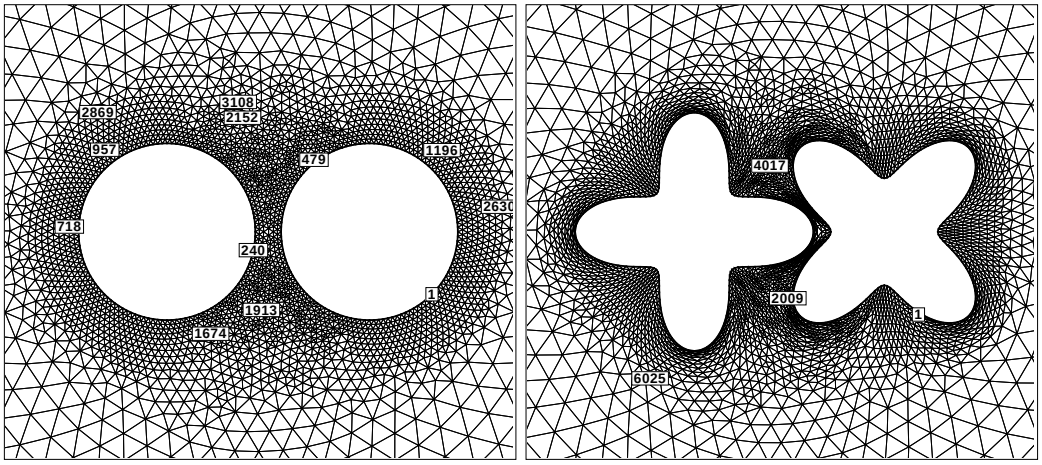


(a) (b)

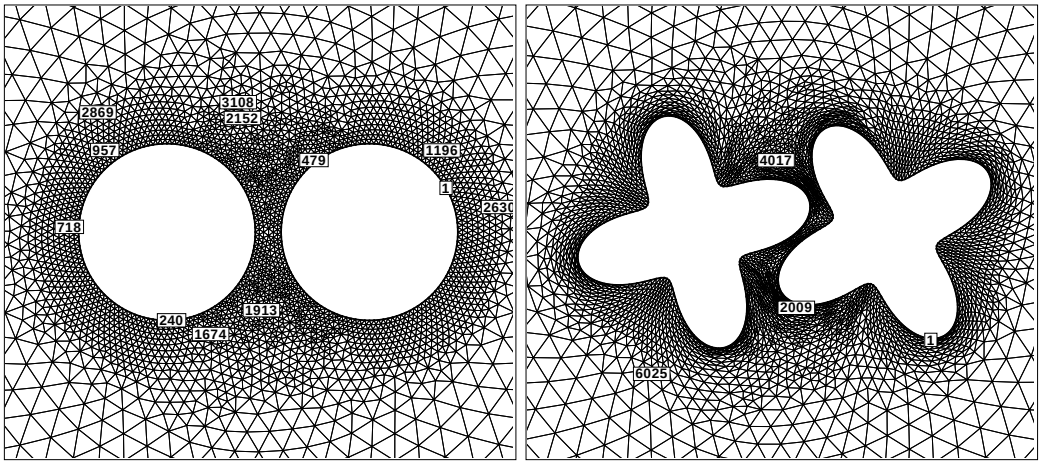
Figure 5.24 Boundaries in (a) computational and (b) physical spaces



(a) Computational Space, t_1 (b) Physical space, t_1



(c) Computational Space, t_{15} (d) Physical space, t_{15}



(e) Computational Space, t_{110} (f) Physical space, t_{110}

Figure 5.25 Sliding the cells around the boundaries in computational and physical domain at three different steps

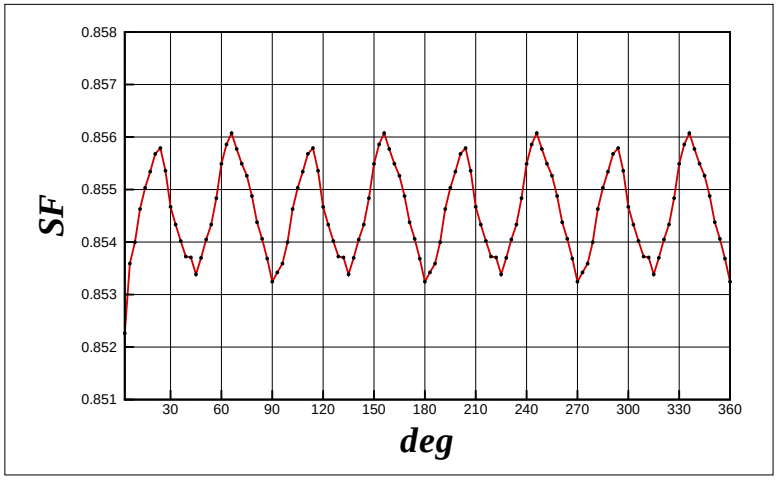


Figure 5.26 SF values for 360^o rotary motion of two four-petal roses

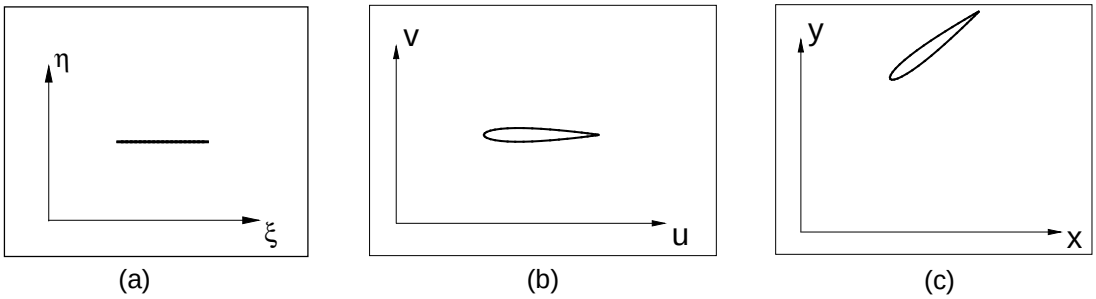


Figure 5.27 (a) Computational, (b) parametric and (c) physical spaces used for combination of translational and rotational motion

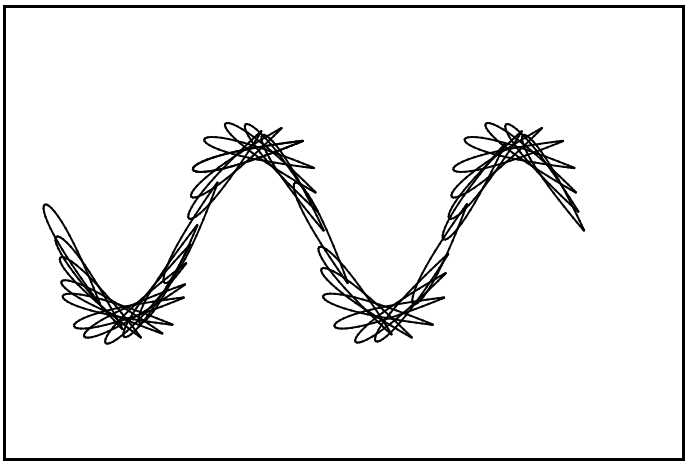
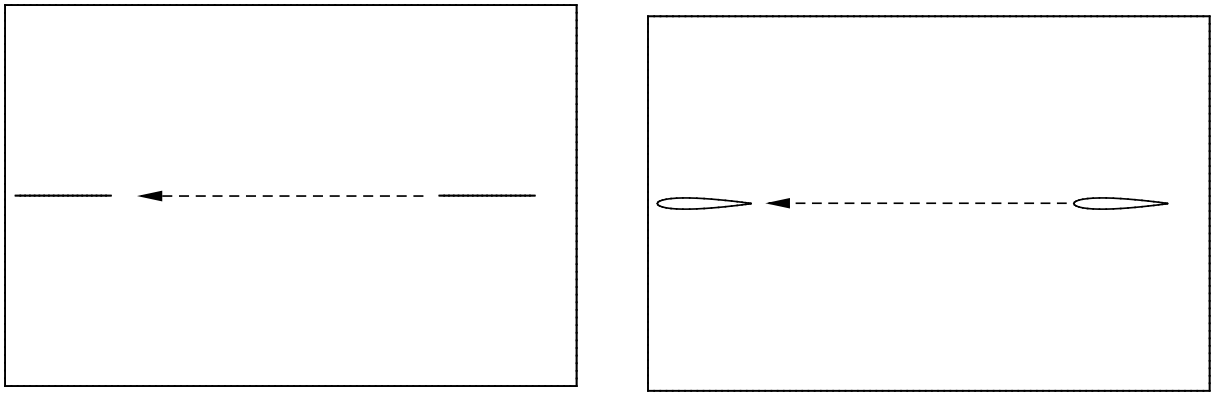
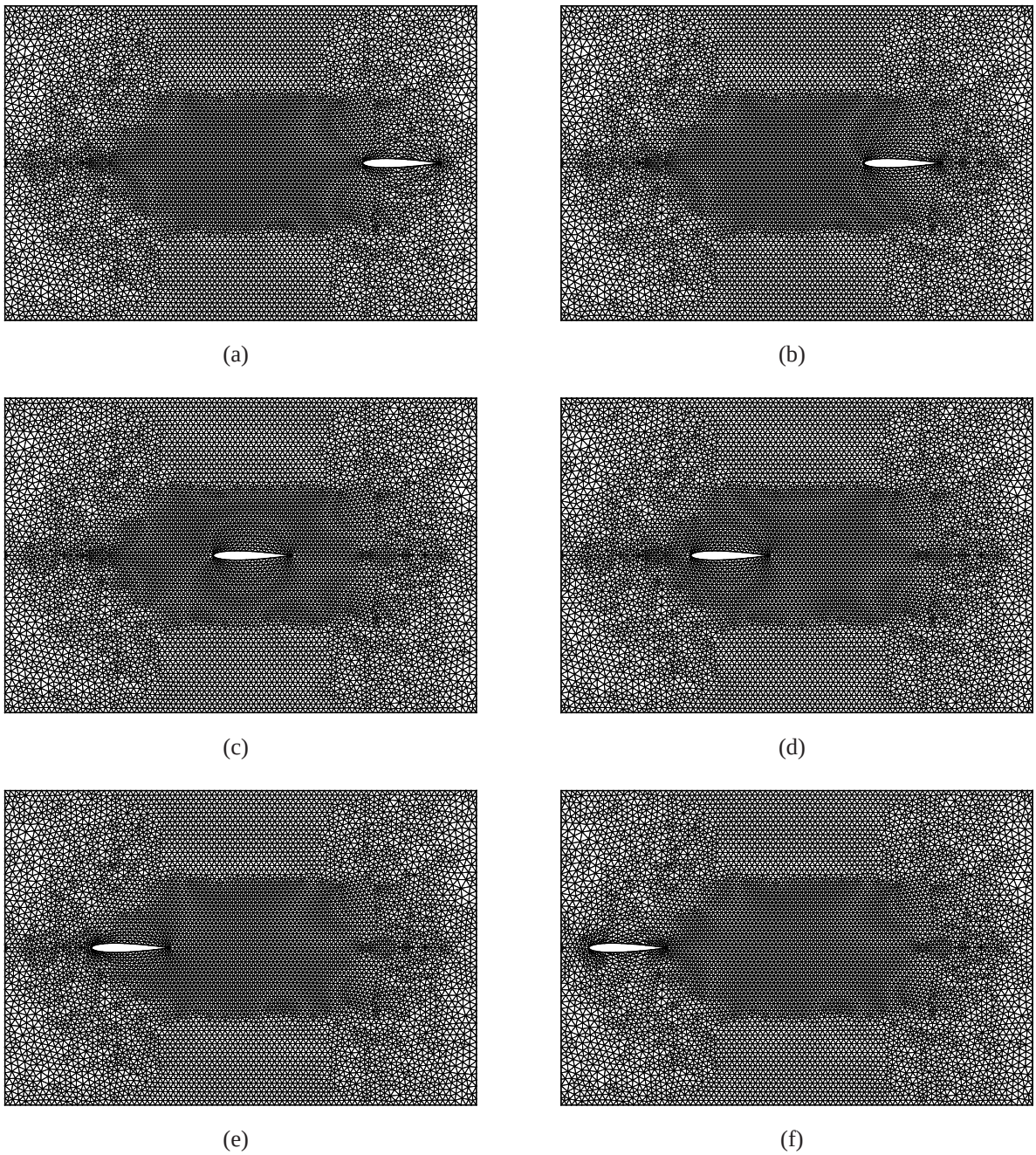


Figure 5.28 Sinusoidal trajectory in physical domain



(a) (b)

Figure 5.29 Linear motion of (a) the slit in the computatioal space and (b) the NACA0012 airfoil in the parametric space



(a) (b)

(c) (d)

(e) (f)

Figure 5.30 Linear motion of a NACA0012 airfoil in the parametric space

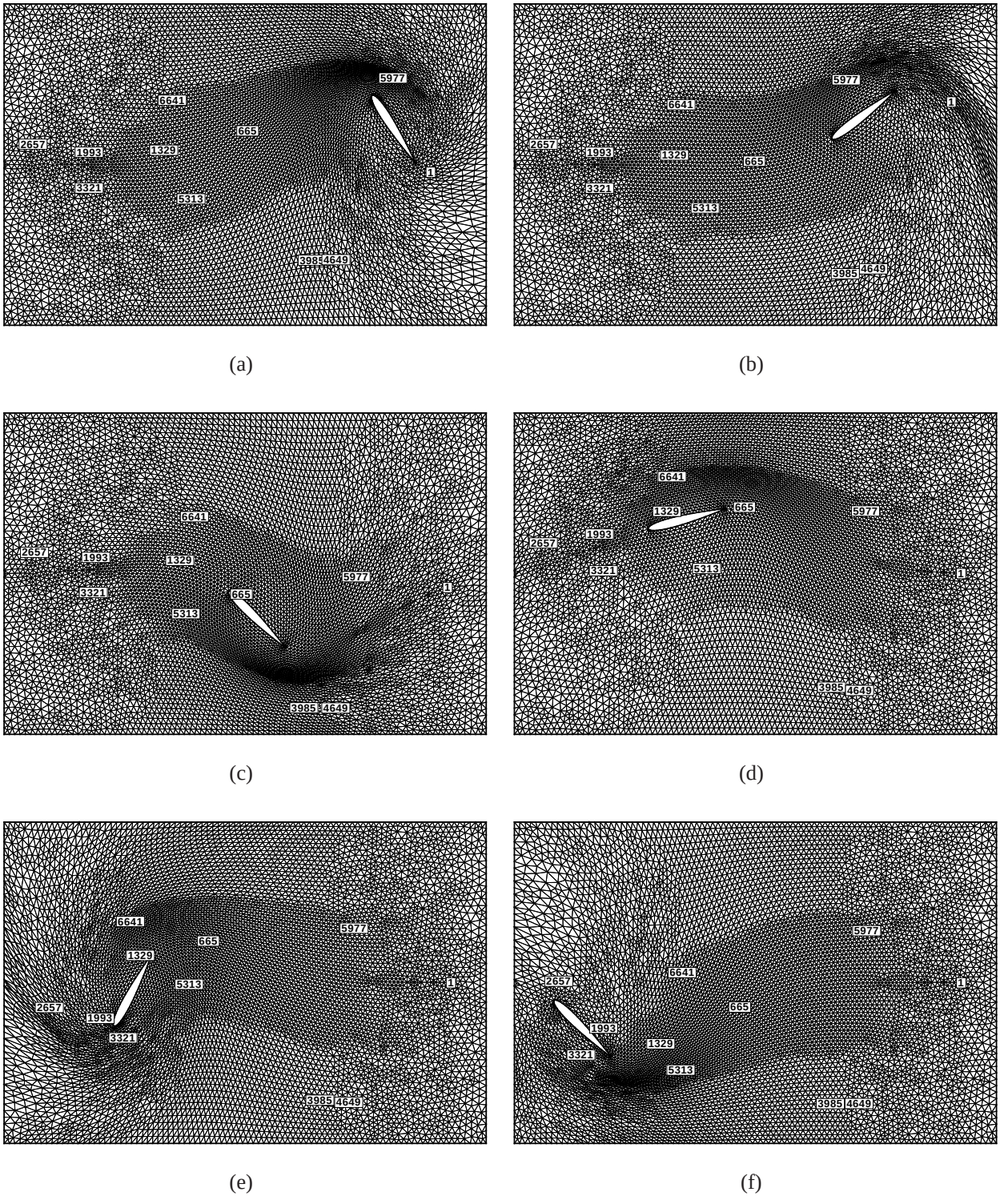


Figure 5.31 Sinusoidal motion of a NACA0012 airfoil in the physical space obtained by sliding mesh and RBF method as a secondary mesh motion technique

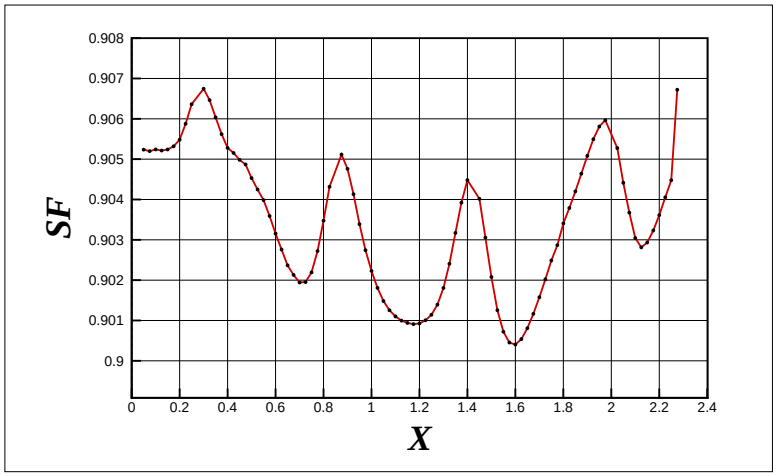


Figure 5.32 SF values for sinusoidal motion of a NACA0012 airfoil

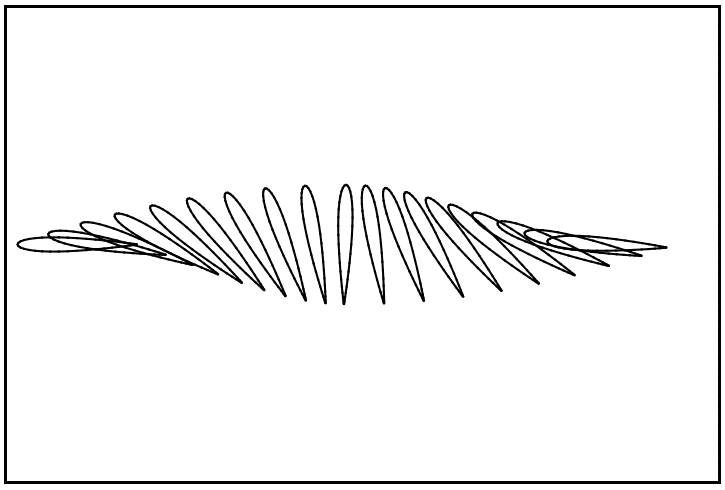


Figure 5.33 Cobra-like trajectory for a NACA0012 airfoil in physical domain

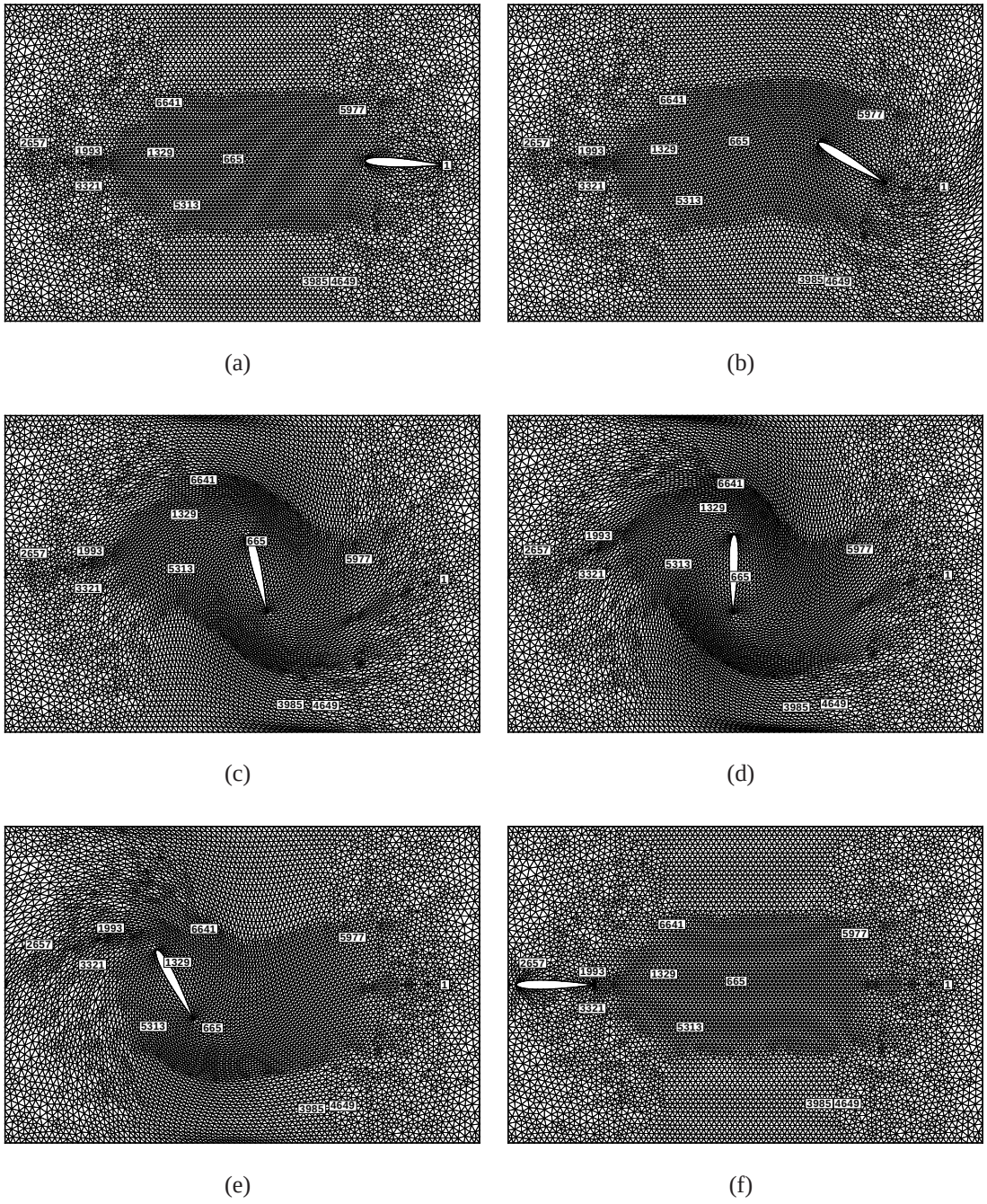


Figure 5.34 Cobra-like motion of a NACA0012 airfoil in the physical space using sliding cells and RBF method as a secondary mesh motion technique

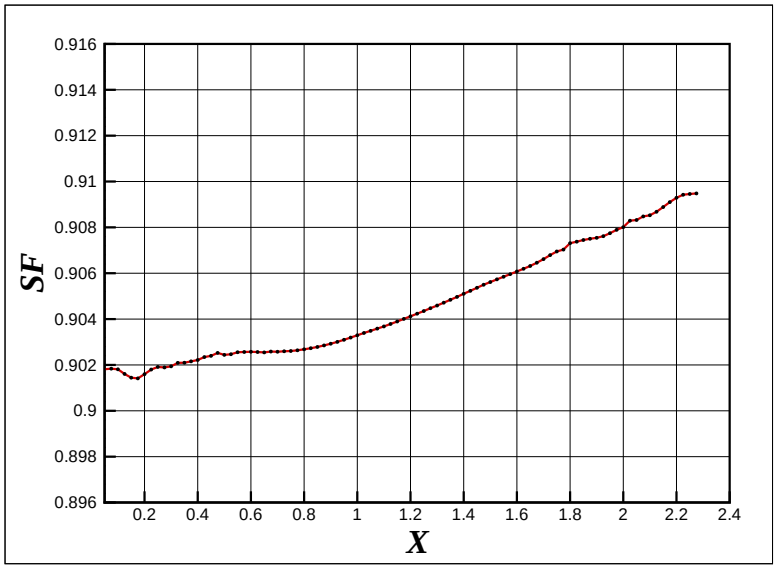


Figure 5.35 SF values for Cobra-like motion of a NACA0012 airfoil

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

In this thesis a novel method for large rigid body motion based on the sliding cells and mapping domains was presented. This method was proposed to overcome the difficulties encountered in traditional mesh motion techniques which maintain the cells connectivity in unstructured grids.

Amongst several methods reviewed, linear elasticity and RBF techniques are considered the most robust tools capable of dealing with large body motions with fixed topology. These methods appear to be promising steps towards moving mesh even with high aspect ratio cells around the boundaries. However, despite the applicability of these two methods, they finally fail in large relative motions or amplitudes. This is because of the fact that the cells remain attached to the boundaries as motion evolves. Our strategy consists in lifting this constraint by allowing the cells to slide over the boundaries, thus avoiding piling up of cells in the physical domain. The major features of this new moving mesh framework are described below.

Maintaining cells connectivity: Sliding cells around the boundaries allows the cells connectivities to remain constant. This sliding technique avoids interpolation errors in moving mesh problems, which has a direct influence on the accuracy and efficiency of the simulations.

Sliding cells in computational space: The major reason of using the computational space in the field of CFD is to ease the discretization process. But the major aim of using a computational mesh in this thesis is to ease the management of motion. In this approach, the boundaries are simplified for translation and rotational motions, when the sliding technique will not suffer from rapid changes in boundary curvature, and also the problem of leading edge in bluff bodies. Also, the cells in computational space always remain isotropic, therefore, there is no limit to motion due to the piling up and skewess of neither cells in the computational nor physical domains.

Moreover, since the cells slide over the boundaries as much as one boundary edge at each motion step, GCL can be applied on the physical mesh currectly. In other words, the time rate of change of the total volume of the physical domain remains constant.

Mapping operators: To transfer the computational mesh to physical space at each step, various mapping equations were explored and applied. Amongst these approaches, two types of operators, the Winslow and the AO functional, have been ultimately chosen to transfer the mesh from computational to physical space. These nonlinear operators have been solved on unstructured grids around different geometries. We show, according to the type of equations that the Winslow operator gives the smoothest meshes, whereas the AO functional deals better with rapid changes in curvature and nonsmooth physical boundaries. Therefore, choosing a correct mapping operator depends on the geometric complexity, whereas, the type of motion does not have a significant role in this matter.

In addition, two new approaches to discretize the mapping operators, one based on the finite difference and the other on finite volume, have been considered. The former one is more efficient while the latter method is more robust. However, for thin physical boundaries in translation motion, the finite difference scheme is suggested because of its efficiency.

Mesh smoothness vs mesh quality: Although, the definition of a good mesh is usually related to the control of solution error, the initial mesh smoothness and quality play a significant role in increasing the efficiency and accuracy of simulations. Many measures have been introduced in the literature to evaluate the size, shape, skewness and mesh quality. Nonetheless, these methods are unable to measure the mesh smoothness. In this work a new method is presented to measure the global smoothness of a mesh. It can be generally said that there is a link between quality and smoothness, however, for some cases they may be contradictory. In such cases, despite the fact that we may have elements with acceptable quality, the shape of cells are dispersed. This usually occures when the mapping operators are nonelliptic.

Combination of mesh sliding and mesh deformation techniques: Finally, the RBF method as a secondary mesh motion technique in physical space and after sliding procedure was used. This new combined approach improves the capability of the method to handle curvilinear trajectories. The results for this approach showed very smooth physical meshes for arbitrary motion trajectories. Furthermore the meshes near the boundaries did not show any evidence of severe

deformation.

6.2 Future work

The following extensions are recommended in order to achieve a more robust mesh motion technique for large amplitudes and fixed connectivity.

- Improvement to the finite volume discretization technique for mapping operators. Using a marching technique to solve parabolic PDEs is more adapted with these type of operators.
- Implementing an optimization technique to optimize the values of $\omega_L,\ \omega_O$ and ω_A in order to achieve the best combination of functionals.
- Finding a meaningful definition for CFL_{mesh} and Δt_{mesh} for moving meshes. This has led us to preserve the grid validity due to the moving boundary.
- Adding viscous layers to the sliding procedure.
- Extending the mapping and sliding procedures to both 3D structured and unstructured meshes.

REFERENCES

Alauzet, F. & Olivier, G. (2011). A new changing-topology ALE scheme for moving mesh unsteady simulations. In *49th AIAA Aerospace Sciences Meeting and Exibit*, AIAA-2011-0474. Orlando, FL.

Anderson, A., Zheng, X., & Cristini, V. (2005). Adaptive unstructured volume remeshing -i: The method. *Journal of Computational Physics*, *208*, 616–625.

Azarenok, B. (2009). Generation of structured difference grids in two-dimensional nonconvex domains using mappings. *Journal of Computational Mathematics and Mathematical Physics*, *49*(5), 797–809.

Baker, T. J. (2001). Mesh movement and metamorphosis. In *Proceedings of the 10th International Meshing Roundtable*, (pages 387–396). Newport Beach, CA: Sandia National Laboratory.

Baker, T. J. (2003). Adaptive modification for time evolving meshes. *Journal of Materials Science*, *38*, 4175–4182.

Batina, J. (1991). Unsteady Euler algorithm for unstructured dynamic mesh for complex-aircraft aerodynamic analysis. *AIAA Journal*, *29*(3), 327–333.

Benek, J., Steger, J., Dougherty, F., & Buning, P. (1986). Chimera: A grid-embedding technique. Technical Report AEDC-TR-85-64, Arnold Engineering Development Center.

Blom, F. (2000). Consideration on the spring analogy. *International Journal of Numerical Methods in Fluids*, *32*(6), 4175–4182.

Castillo, J., Steinberg, S., & Roache, P. (1988). On the folding of numerically generated grids: use of reference grid. *Journal of Communications in Applied Numerical Methods*, *4*, 471–481.

Chibisov, D., Ganzha, V., Mayr, E., & Vorozhtsov, E. (June 2006). On the provable tight approximation of optimal meshing for non–convex regions. In *Computational topology and geometry in application of computer algebra (ACA2006)*. Varna, Bulgaria.

Cristini, V., Blawzdziewicz, J., & Loewenberg, M. (2001). An adaptive mesh algorithm for evolving surfaces: Simulations of of drop breakup and coalescence. *Journal of Computational Physics*, *168*, 445–463.

Dwight, R. (2006). Robust mesh deformation using the linear elasticity equations. In *4th International Conference on Computational Fluid Dynamics, ICCFD4*, (pages 401–406). Ghent, Belgium.

Farhat, C., Degand, C., Koobus, B., & Lesoinne, M. (1998). Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Journal of Computational Methods in Applied Mechanics and Engineering*, *63*(1-4), 231–245.

Farhat, C. & Lesoinne, M. (1996). Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations. *Journal of Computer Methods in Applied Mechanics and Engineering*, *134*(1-2), 71–90.

Garon, A. (1983). Génération de maillages sur des surfaces et résolution numérique des équations de navier-stokes dans une cascade. Master's thesis, École polytechnique de Montréal.

Hassan, O., Probert, E., & Morgan, K. (1998). Unstructured mesh procedures for the simulation of three-dimensional transient compressible inviscid flows with moving boundary components. *International Journal for Numerical Methods in Fluids*, *27*, 41–55.

Hassan, O., Probert, E., Morgan, K., & Weatherill, N. P. (2000). Unsteady flow simulation using unstructured meshes. *Computer Methods in Applied Mechanics and Engineering*, *189*, 1247–1275.

Hassan, O., Sorensen, K. A., Morgan, K., & Weatherill, N. P. (2007). A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing. *International Journal for Numerical Methods in Fluids*, *53*(8), 1243–1266.

Helenbrook, B. T. (2001). Mesh deformation using the biharmonic operator. *International Journal for Numerical Methods in Engineering*, *56*(7), 1–30.

Illinca, A., Camarero, R., Trépanier, J.-Y., & Reggio, M. (1995). A new adaptive technique using moving grids. *33rd Aerospace Sciences Meeting and Exhibit, AIAA-95-0669*.

Karman, S. (2010). Virtual control volumes for two dimensional unstructured elliptic smoothing. In *Proceedings of the 19th International Meshing Roundtable*, (pages 121–142). Chattanooga, TN: Sandia National Laboratory.

Khamayseh, A. & Hansen, G. (2000). Quasi-orthogonal grids with impedance matching. *SIAM Journal of Scientific Computing*, *22*, 1220–1237.

Khamayseh, A. & Mastin, C. (1996). Computational conformal mapping for surface grid generation. *Journal of Computational Physics*, *123*, 394–401.

Khattri, S. (2007). Grid generation and adaptation by functionals. *Journal of Computational and Applied Mathematics*, *26*(2), 235–249.

Knupp, P. & Steinberg, S. (1993). *Fundamentals of Grid generation*. CRC Press, ISBN 0-8493-8987-9.

Knupp, P. M. (1999). Winslow smoothing on two-dimensional unstructured meshes. *Engineering with Computers*, *15*(3), 263–268.

Kovalev, K. (2005). Unstructured hexahedral non-conformal mesh generation. Master's thesis, Vrije Universiteit Brussel.

Kwak, S. & Pozrikidis, C. (1998). Adaptive triangulation of evolving, closed, or open surfaces by the advancing-front. *Journal of Computational Physics*, *145*, 61–88.

Lin, P., Martinelli, L., Baker, T., & Jameson, A. (2001). Two-dimensional implicit time dependent calculations for incompressible flows on adaptive unstructured meshe. In *15th AIAA Computational Fluid Dynamics Conference*, AIAA-2001-2655, Anaheim, CA., USA.

Persson, P.-O. (2005). Size functions and mesh generation for high-quality adaptive remeshing. In *3rd MIT conference on computational fluid and solid mechanics*, (page 13).

Rendall, T. & Allen, C. (2008). Unified fluid-structure interpolation and mesh motion using radial basis functions. *International Journal for Numerical Methods in Engineering*, *74*, 1519–1559.

Rendall, T. & Allen, C. (2009). Efficient mesh motion using radial basis functions with data reduction algorithms. *Journal of Computational Physics*, *228*, 6231–6249.

Rhee, S. (2005). Unstructured grid based Reynolds averaged Navier-Stokes method for liquid tank sloshing. *Transactions of the ASME - Journal of Fluids Engineering*, *127*(3), 572–582.

Rhee, S. (2009). Unsteady Reynolds averaged Navier-Stokes method for free-surface wave flows around surface-piercing cylindrical structures. *Journal of Waterway, Port, Costal and Ocean Engineering*, *135*(4), 139–143.

Saksono, P. H., Dettmer, W., & Peric, D. (2007). An adaptive remeshing strategy for flows with moving boundaries and fluid structure interaction. *International Journal for Numerical Methods in Engineering*, *7*(9), 193–218.

Schneider, R., Oberschelp, W., Kopp, R., & becker, M. (1992). New and effective remeshing scheme for the simulation of metal forming processes. *Engineering With Computers*, *8*, 163–176.

Spekreijse, S. (1995). Elliptic generation based on laplace equations and algebraic transformations. *Journal of Computational Physics*, *118*, 38–61.

Spekreijse, S. (1996). Elliptic generation systems. Technical Report NLR TP 96735, National Aerospace Laboratory NLR.

Stein, K., Tezduyar, T., & Benney, R. (2003). Mesh moving techniques for fluid-structure interactions with large displacements. *Journal of Applied Mechanics*, *70*, 58–63.

Trépanier, J.-Y., Reggio, M., Paraschivoiu, M., & Camarero, R. (1993). Unsteady euler solutions for arbitrarily moving bodies. *AIAA Journal*, *31*(10), 1869–1876.

Villamizar, V. & Acosta, S. (2009). Elliptic grids with nearly uniform cell area and line spacing. *Electronic Transactions on Numerical Analysis*, 59–75.

Villamizar, V. & Weber, M. (2007). Boundary-conforming coordinates with grid line control for acoustic scattering from complexly shaped obstacles. *Wiley InterScience*, (20235).

Winslow, A. M. (1967). Numerical solution of the quasilinear Poisson equations in a nonuniform triangle mesh. *Journal of Computational Physics*, *2*, 149–172.

Winslow, A. M. (1997). Numerical solution of the quasi-linear Poisson equation in a non-uniform triangular mesh. *Journal of Computational Physics*, *135*, 128–138.

Yang, Z. & Mavriplis, D. (2005). Unstructured dynamic meshes with higher-order time integration schemes for the unsteady Navier–Stokes equations. In *Proceedings of the 41th AIAA Aerospace Sciences Meeting and Exhibit*, 2005-1222. Reno, NV.

Yang, Z. & Mavriplis, D. J. (2007). Mesh deformation strategy optimized by the adjoint method on unstructured meshes. *AIAA Journal*, *45*(12), 2885–2896.

Zhang, H., Reggio, M., Trépanier, J.-Y., & Camarero, R. (1993). Discrete form of the GCL for moving meshes and its implementation in CFD schemes. *Computers and Fluids*, *22*(1), 9–23.