

Direct structured finite element mesh generation from three-dimensional medical images of the aorta

Sharareh Bayat

Supervisors:

Dr. Michel R. Labrosse

Dr. Dan S. Neculescu

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mechanical Engineering

Ottawa-Carleton Institute for Mechanical and Aerospace Engineering

University of Ottawa

© Sharareh Bayat, Ottawa, Canada, 2014

Abstract

Three-dimensional (3-D) medical imaging creates notable opportunities as input toward engineering analyses, whether for basic understanding of the normal function or pathophysiology of an organ, or for the simulation of virtual surgical procedures. These analyses most often require finite element (FE) models to be constructed from patient-specific 3-D medical images. However, creation of such models can be extremely labor-intensive; in addition, image processing and mesh generation are often operator-dependent, lack robustness and may be of suboptimal quality.

Focusing on the human aorta, the goal of the present work is to create a fast and robust methodology for quadrilateral surface and hexahedral volume meshing from 3-D medical images with minimal user input. By making use of the segmentation capabilities of the 3-D gradient vector flow field combined with original ray-tracing and orientation control algorithms, we will demonstrate that it is possible to incrementally grow a structured quadrilateral surface mesh of the inner wall of the aorta. The process does not only require minimal input from the user, it is also robust and very fast compared to existing methods; it effectively combines segmentation and meshing into one single effort. After successfully testing the methodology and measuring the quality of the meshes produced by it from synthetic as well as real medical image datasets, we will make use of the surface mesh of the inner aortic wall to derive hexahedral meshes of the aortic wall thickness and of the fluid domain inside the aorta. We will finally outline a tentative approach to merge several structured meshes to process the main branches of the aorta.

Acknowledgments

I owe sincere and earnest appreciation to my supervisor Dr. Michel Labrosse, whose deep knowledge and proficiency as well as continuous confidence, invaluable assistance and enthusiasm, made it possible to complete this study. I express my deepest gratitude for his encouragements and constant support. I gratefully acknowledge the funding sources that I received. I express my warmest thankfulness to my co-supervisor Dr. Dan Neculescu, who fully supported and encouraged me in all aspects of my study and academic affairs with enlightening guidance. It has been an honor for me to be their student.

I would also like to gratefully acknowledge my prodigious advisors Dr. Davide Spinello and Dr. Jurek Sasiadek for their bright guidance, insightful comments and assistance along the journey to complete this work. Special thanks also to Dr. Alejandro Guzmán De León for his valuable inputs and advice.

In addition, I wish to express my gratitude to my husband, Mahyar, for his enduring love and tremendous encouragements and companionship. He was always there cheering me up. I am grateful and indebted forever to my parents, Zahra and Hossein, for their endless love and support, without whom I could not have made it here. I would like to thank my sister, Shahrzad, for her encouragements and my parents-in-law, Pouran and Hashem, for their moral supports.

My time at the University of Ottawa was made enjoyable and enriched in large part due to the many friends and colleagues; Thank you Elisha, Ehsan and Mostafa and many more.

I would like to dedicate this thesis to my husband, Mahyar
and my parents, Zahra and Hossein.

Table of Contents

Abstract.....	i
Acknowledgments	ii
1 Introduction	1
1.1 Context of the study	1
1.2 Motivation and objectives.....	4
1.3 Potential applications.....	5
1.4 Framework and outline of this work.....	5
1.5 Contributions.....	7
2 Literature review	9
2.1 Anatomy and common diseases of the aorta.....	9
2.1.1 Aortic aneurysms	11
2.2 Segmentation methods for the cardiovascular system.....	14
2.2.1 Image-based methods.....	15
2.2.1.1 Thresholding methods	15
2.2.1.2 Region growing method	16
2.2.1.3 Classifier methods	18
2.2.1.4 Clustering methods.....	18
2.2.1.5 Artificial neural networks.....	19
2.2.1.6 Markov random field models (MRF) / Statistical pattern recognition....	20
2.2.1.7 Edge tracing.....	20
2.2.2 Model-based methods:	21
2.2.2.1 Atlas-guided segmentation.....	21

2.2.2.2	Deformable models	22
2.2.3	Mathematical development of parametric active contours	28
2.2.3.1	Gradient potential force.....	31
2.2.3.2	Distance potential force.....	32
2.2.3.3	Pressure force	33
2.2.3.4	Gradient vector flow force	34
2.3	Mesh generation techniques	36
2.3.1	Structured mesh	36
2.3.2	Unstructured mesh	38
2.3.3	Structured as opposed to unstructured mesh.....	39
2.4	Existing FE mesh generation techniques from medical images	41
2.4.1	Early mesh-matching techniques	42
2.4.2	Ad-hoc techniques	44
2.4.3	Iso-surface techniques.....	45
2.4.4	Morphing techniques	45
2.4.5	Deformable-model based techniques.....	47
2.4.6	Existing software packages.....	49
2.5	Engineering analysis of the aorta.....	49
2.5.1	Solid mechanics analyses.....	50
2.5.2	Fluid mechanics analyses.....	50
2.5.3	Fluid-structure interaction analyses	50
3	Three-dimensional gradient vector flow.....	52
3.1	Formulation	52

3.2	Implementation of 3-D GVF field using synthetic images.....	54
3.3	Implementation of 3-D GVF field using medical images.....	64
3.3.1	Canny-edge detection.....	64
3.3.2	GVF field for CT images	65
4	Direct finite element implementation and a variant.....	70
4.1	Creation of an arbitrary structured mesh	70
4.2	Finite element formulation	73
4.2.1	Derivation of the local stiffness matrix for a space frame element	73
4.2.2	Using the 3-D GVF field as a force field.....	79
4.2.3	Results.....	80
4.2.4	Discussion	84
4.3	Using the 3-D GVF as an imposed displacement field	85
4.3.1	Results.....	87
4.3.2	Discussion	89
5	Automatic surface mesh generation.....	90
5.1	Methodology	91
5.2	Initialization.....	94
5.3	Ray-tracing method.....	96
5.4	Marching method	100
5.5	Orientation control algorithm.....	103
5.6	Preliminary structured mesh	107
5.7	Structured mesh refinement.....	108
5.8	Test on the Visible Human dataset	111

5.9	Test on CT dataset with descending aortic aneurysm	117
5.10	Test on CT dataset with thoracic aortic aneurysm	122
5.11	Discussion of results	126
5.12	Output to third-party FE code.....	128
6	Evaluation of surface mesh quality.....	132
6.1	Reconstruction accuracy.....	132
6.1.1	Visual inspection.....	132
6.1.2	Phantom creation	134
6.2	Mesh quality inspection	137
7	Volume mesh generation.....	148
7.1	Aortic wall volume mesh generation	148
7.2	Fluid domain volume mesh	151
8	Mesh merging for aortic branches.....	158
8.1	Individual branch meshing.....	159
8.2	Bilinear patches and decomposition of transition region	161
8.3	Meshing of transition regions.....	165
8.4	Coon's patches	167
9	Discussion	169
9.1	Significance and novelty of the approach	169
9.2	Limitations	170
9.3	Recommendations for future work.....	171
9.3.1	Finalize volume meshing capabilities	171
9.3.2	Develop and finalize branch merging capabilities.....	171

9.3.3	Investigate segmentation tools other than the 3-D GVF	171
9.3.4	Considering other aortic datasets and other organs	172
9.3.5	Toward a complete mesh generation package	172
10	Conclusion	173
	Appendix A: 3-D GVF computation	174
	Appendix B: Bicubic Coon's patch	177
	Bibliography	179

List of Figures

Figure 1.1: Typical workflow for patient-specific geometry modeling toward finite element analysis. The steps included in the red box are time consuming.	2
Figure 2.1: Left: the entire aorta and its major branches (WebMD, 2009) ; Middle: The heart, aortic arch and aortic valve (WebMD, 2013); Right: the aortic arch and its branches, with the brachiocephalic trunk (left), the left common coronary artery (middle) and left subclavian artery (right) (Photograph taken in Dr. Labrosse’s lab of an excised human aorta).....	9
Figure 2.2: Left: Planar simplification of the aortic arch introduced by (Gao et al., 2005); Right: A 3-D non-planar model developed by (Fujioka et al., 2000).....	11
Figure 2.3: Preoperative computed tomography reconstruction of a 7.4 cm abdominal aortic aneurysm (Metcalfé et al., 2011).	13
Figure 2.4: Thresholding a grayscale image: (a) Original image, (b) Thresholded image.	16
Figure 2.5: Region Growing, (a): Start of growing region, (b): Growing process after iteration (Marshall, 2008).	17
Figure 2.6: Joint approach: (a), (c), (e) Segmentation results of the 3-D joint approach for three 3-D CTA images showing a pathology. (b), (d), (f) Vessel contours using the model-based approach (blue) and the 3-D joint approach (red) for a section of the 3-D CTA images (Biesdorf et al., 2012).	25
Figure 2.7: Resulting 3-D model of abdominal aorta (Loncaric et al., 2000).....	27
Figure 2.8: (a) Convergence of active contour using (b) gradient potential force, (c) close-up within the boundary concavity (Xu & Prince, 1998).....	31
Figure 2.9: (a) Convergence of active contour using (b) Distance potential force, (c) close-up within the boundary concavity (Xu & Prince, 1998).....	33

Figure 2.10: Gradient vector flow, Left: deformation of snake with GVF forces. Middle: GVF external forces. Right: close-up within the boundary concavity (Xu & Prince, 1998)..	35
Figure 2.11: Structured mesh schematic.....	37
Figure 2.12: Examples of O- and C-grids (Innovative-cfd, 2011)	38
Figure 2.13: Representation of a structured mesh in a cylindrical model.	39
Figure 2.14: Geometry correspondence and morphing: a) Surface of the target model, b) Comparison of coronal (center) and sagittal (right) cross sections of the target model (red line), (Sigal et al., 2008).	46
Figure 3.1: MATLAB code to create a synthetic volume image of the aorta.....	55
Figure 3.2: MATLAB code for 3-D GVF field computation.	57
Figure 3.3: Implementation of 3-D GVF on synthetic aortic images; (a), (e), (i); Different cross sections of the synthetic images; (b), (f), (j) GVF field overlaid on top of the original images; (c), (d), (g), (h), (k), (l) Magnified GVF field overlaid on top of the original images-magnified, after 8 iterations, with $\mu=0.2$	60
Figure 3.4: Implementation of 3-D GVF on synthetic images; (a), (e), (i); Different cross sections of the synthetic images; (b), (f), (j) GVF field overlaid on top of the original images; (c), (d), (g), (h), (k), (l) Magnified GVF field overlaid on top of the original images-magnified, after 15 iterations with $\mu=0.2$	63
Figure 3.5: Anatomical planes.....	65
Figure 3.6: Left: top: Original image; middle and bottom: Gradient image; Right: top: Edge map; middle and bottom: GVF field. (a) Axial, (b) Coronal, (c) Sagittal.	68
Figure 4.1: Pseudo-code (in MATLAB format) for initial cylindrical structured mesh generation.	71
Figure 4.2: Centerline and arbitrary cylindrical structured mesh of the synthetic aorta.....	72
Figure 4.3: Space frame element, adapted from (Bhatti, 2006).....	73

Figure 4.4: Local degrees of freedom for space frame element (Bhatti, 2006).	74
Figure 4.5: (a) Initial mesh (black), and deformed mesh (red) using 3-D GVF field as applied force, after 36 iterations, (b): Magnified deformed mesh.	81
Figure 4.6: Location of original initial nodes (black) and deformed nodes (red), at (a): Cross section #19, (b): Cross section #12 and (c): Cross section #2, after 36 iterations.	82
Figure 4.7: Initial mesh (black), and deformed mesh (red) using 3-D GVF field as applied force after 138 iterations.	83
Figure 4.8: Locations of original initial nodes (black) and deformed nodes (red), at (a): Cross section #19, (b): Cross section #12 and (c): Cross section #2.	84
Figure 4.9: Mesh deformation algorithm based on 3-D GVF field interpreted as imposed displacement.	86
Figure 4.10: a) Original mesh (black), and deformed mesh (red) using 3-D GVF field interpreted as imposed displacement field, b) magnified deformed mesh.	87
Figure 4.11: Locations of original initial nodes (black) and deformed nodes (red), at (a): Cross section #19, (b): Cross section #12 and (c): Cross section #2.	88
Figure 5.1: (a): Original image volume from a CT dataset, (b): Region of interest (ROI).	91
Figure 5.2: Surface mesh generation algorithm.	93
Figure 5.3: 3-D GVF field for synthetic aortic image obtained after 6 iterations and a few seconds of computation.	94
Figure 5.4: (a) Selection of first cross-section, (b) Selection of second cross-section to direction for mesh generation.	95
Figure 5.5: Partial illustration of the ray-tracing algorithm for cross-section k at iteration i . The 3-D GVF field is not represented for clarity.	96
Figure 5.6: Abridged ray-tracing algorithm (in MATLAB format).	99

Figure 5.7: Ray-tracing method application on cross-section k	100
Figure 5.8: Partial illustration of the marching algorithm to image and mesh cross-section $k + 1$ starting from cross-section k . The 3-D GVF field is not represented for clarity.	101
Figure 5.9: Coordinate system and rotation angles.....	102
Figure 5.10: Orientation control schematic.	104
Figure 5.11: Marching algorithm with orientation control (in MATLAB format).....	106
Figure 5.12: Progression of mesh generation process on synthetic aorta.....	107
Figure 5.13: Abridged mesh refinement algorithm (in MATLAB format).	110
Figure 5.14: Final mesh for the synthetic aorta.	111
Figure 5.15: An original image from the Visible Human dataset (U.S. National Library of Medicine, 2013).....	112
Figure 5.16: 3-D view of the cropped volume for the Visible Human dataset.....	113
Figure 5.17: 3-D GVF field after 20 iterations, with $\mu = 0.1$	113
Figure 5.18: Steps of the preliminary mesh generation for the Human Visible dataset.	114
Figure 5.19: The effect of varying n ; top: $n = 15$, bottom: $n = 6$	115
Figure 5.20: Final surface mesh for the Human Visible dataset.....	116
Figure 5.21: Final surface mesh for the Human Visible dataset. (a): 100 nodes in circumference and 60 cross-sections, (b): 50 nodes in circumference and 70 cross- sections.	117
Figure 5.22: top: 3-D view of ROI, bottom: 3-D GVF field after 20 iterations, with $\mu = 0.1$	118
Figure 5.23: Steps of the preliminary mesh generation for dataset with descending aortic aneurysm.....	119
Figure 5.24: The effect of varying n ; left: $n = 6$ (no folding); right: $n = 15$ (folding).....	120
Figure 5.25: Final surface mesh with 60 nodes in circumference and 60 cross-sections.	121

Figure 5.26: Final surface mesh: (a): 80 nodes in circumference and 60 cross-sections, (b): 80 nodes in circumference and 80 cross-sections.	121
Figure 5.27: top: 3-D view of the ROI, bottom: 3-D GVF field after 20 iterations, with $\mu=0.1$	123
Figure 5.28: Steps of the preliminary mesh generation for dataset with thoracic aortic aneurism.....	124
Figure 5.29: The effect of varying n ; left: $n=10$, right: $n=11$	125
Figure 5.30: Final surface mesh with 60 nodes in circumference and 60 cross-sections.	125
Figure 5.31: Final surface mesh: (a) 20 nodes in circumference and 100 cross-sections, (b) 60 nodes in circumference and 100 cross-sections.	126
Figure 5.32: Quadrilateral element connectivity table creation algorithm.	129
Figure 5.33: Quadrilateral element node allocation process.....	129
Figure 5.34: LS-DYNA mesh for synthetic aorta.....	130
Figure 5.35: LS-DYNA mesh for the Visible Human dataset.....	130
Figure 5.36: LS-DYNA mesh for descending thoracic aortic aneurysm dataset.....	131
Figure 5.37: LS-DYNA mesh for thoracic aortic aneurysm dataset.....	131
Figure 6.1: Mesh accuracy inspection: Boundary comparison for the synthetic aorta.....	133
Figure 6.2: Phantom (green) and reconstructed (red) meshes embedded in 3-D dataset for the synthetic aorta. Units in mm.	135
Figure 6.3 left: Phantom mesh, right: reconstructed mesh.	136
Figure 6.4: Aspect ratio of a quadrilateral element.	138
Figure 6.5: Skewness angles for a quadrilateral element.....	139
Figure 6.6: Mesh quality inspection for the reconstructed synthetic aorta with 3480 elements, (a): minimum quad angles, (b): minimum quad angles, (c): Aspect ratio, (d): Warpage, (e): Skewness, (f): Jacobian ratio.	142

Figure 6.7: Mesh quality inspection for the reconstructed Visible Human dataset, (a): minimum quad angles, (b): minimum quad angles, (c): Aspect ratio, (d): Warpage, (e): Skewness, (f): Jacobian ratio.	146
Figure 7.1: Surface meshes of the outer wall (magenta wireframe, yellow shade) and inner wall (yellow wireframe, cyan shade) to be used toward hexahedral meshing of the aortic wall in the synthetic aorta dataset.	149
Figure 7.2: Surface meshes of the outer wall (magenta wireframe, yellow shade) and inner wall (yellow wireframe, cyan shade) to be used toward hexahedral meshing of the aortic wall in the Visible Human dataset.	150
Figure 7.3: Surface meshes of the outer wall (magenta wireframe, yellow shade) and inner wall (yellow wireframe, cyan shade) to be used toward hexahedral meshing of the aortic wall in the descending aortic aneurysm dataset.	150
Figure 7.4: Algorithm for creation of square core tube mesh inside fluid domain.	152
Figure 7.5: Core surface mesh inside fluid domain of synthetic aorta.	153
Figure 7.6: Node pattern for hexahedral meshing of the fluid domain, based on the surface mesh of the inner aortic wall and the square core tube. The nodes in black may be used to generate a refined mesh near the aortic wall, while those in green and in the square core may define a coarser mesh.	154
Figure 7.7: Three fundamental surface meshes for the synthetic aorta dataset.	155
Figure 7.8: Three fundamental surface meshes for the Visible Human dataset.	156
Figure 7.9: Three fundamental surface meshes for the descending aortic aneurysm dataset.	157
Figure 8.1: Top to bottom: branch meshing process; Left: left subclavian artery, Middle: left common coronary artery, Right: brachiocephalic trunk.	160
Figure 8.2: Disjointed initial meshes of aorta and branches.	161

Figure 8.3: Boundary points definition for bilinear patch creation. See main text for details.	163
Figure 8.4: Boundary lines drawn for left subclavian artery bifurcation. Units in mm.....	164
Figure 8.5: Boundary lines drawn for left common coronary artery (left) and brachiocephalic trunk (right). Units in mm.....	165
Figure 8.6: Mesh generated using bilinear patch for transition region between aortic arch and left subclavian artery. Left: front view, Right: back view. Units in mm.	166
Figure 8.7: Bilinear Coon's patch approach (adapted from Wolf et al., 2003).	168

List of Tables

Table 2.1: Normal Aortic Dimensions in adults (Goldwin 1990).	10
Table 2.2: 12-month AAA rupture risk by diameter (Moll et al., 2011).	12
Table 4.1 Material and circular cross-section properties for frame elements.....	78
Table 5.1 Voxel dimensions for Visible Human dataset.	112
Table 5.2 Voxel dimensions for thoracic aortic aneurysm dataset.	122
Table 6.1 Quantitative evaluation of mesh reconstruction accuracy for the synthetic aorta, for 58 cross-sections and 60 nodes per cross-section.	137
Table 6.2 Mesh quality summary for the synthetic aorta.....	143
Table 6.3: Mesh quality summary for the Visible Human data set.	147

1 Introduction

1.1 Context of the study

With the recent improvements and widespread use of three-dimensional (3-D) medical imaging techniques such as computerized tomography (CT), magnetic resonance (MR) and echography, many organs can now be visualized in vivo with great resolution and accuracy. While this already helps physicians make better diagnoses of diseases, 3-D imaging also opens the door to engineering simulations, whether for basic understanding of the physiology of the organ, or for simulation of surgical procedures. For example, plastic surgeons can scan a patient's deformed jaw and use engineering simulation tools to determine how much and where material must be added or removed for best effect before the surgery takes place. Similarly, neurosurgeons increasingly use 3-D modeling capabilities to optimize access to specific areas inside the brain. In many of these applications, specific grids or meshes need to be generated to represent the organ surface or volume such that finite element (FE) simulations may be carried out. Patient-specific FE models constructed from 3-D medical images can be used to compensate for brain deformation during neurosurgery (Ferrant et al., 2001), predict deformation of the breast (Azar et al., 2002), estimate deformation of the heart's left ventricle (Papademetris et al., 2002), or register a tumor-bearing brain scan to an anatomical atlas (Kyriacou et al., 1999).

As illustrated in Figure 1.1, a common first step to constructing patient specific meshes from medical images consists in applying some type of image processing technique, such as contrast enhancement or filtering classification. This first step is

essential to improving the quality of the original images such that they can be analyzed more efficiently.

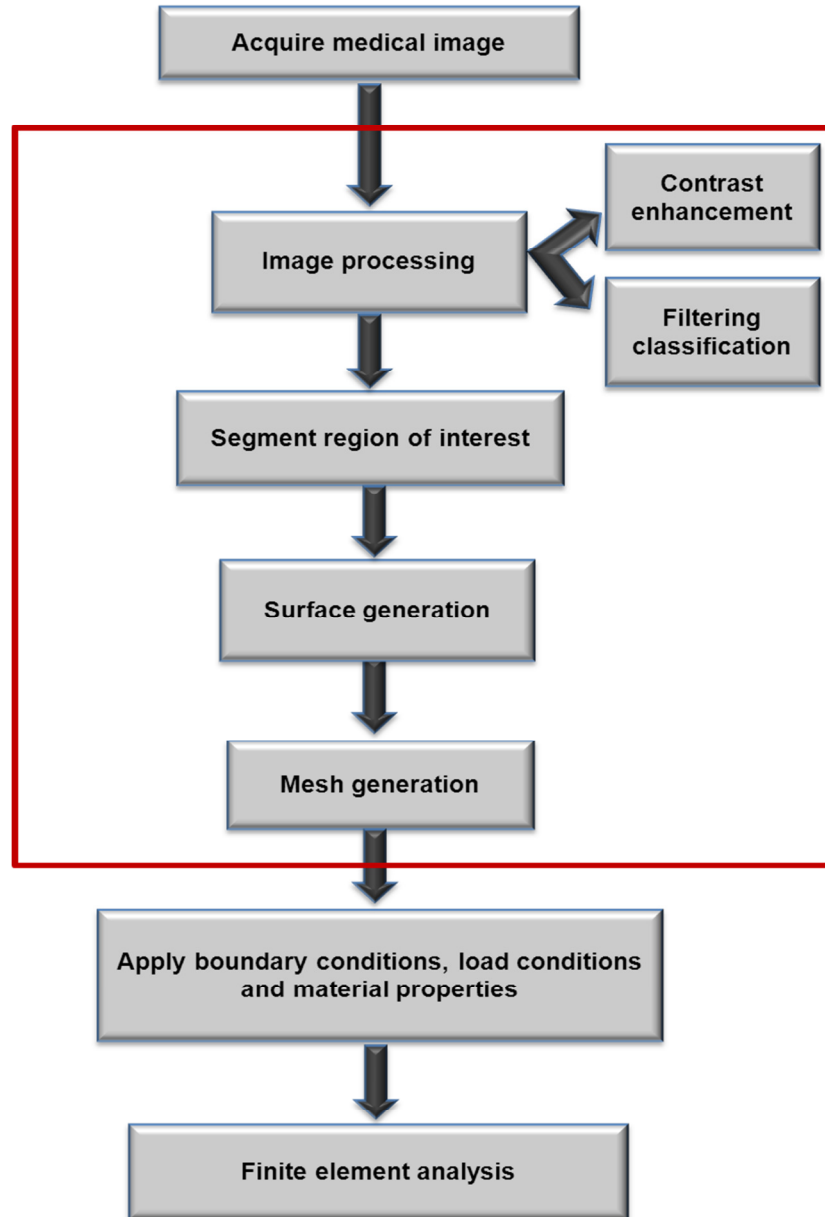


Figure 1.1: Typical workflow for patient-specific geometry modeling toward finite element analysis. The steps included in the red box are time consuming.

The next step usually consists in extracting the surface of the organ by segmentation, which is the determination of the organ's contours. Once the required

surface is reconstructed, different algorithms may be used to generate the FE mesh. For further engineering analysis using the FE method, boundary conditions and material properties need to be assigned to the model.

As can be seen, an important initial step in FE analysis is mesh generation. It involves the discretization of the 3-D domain (herein organ) into small elements of simple geometry, such as tetrahedra or hexahedra (Mohamed & Davatzikos, 2004). Obviously, meshing requires the acquisition of specific geometrical information about the domain to be modelled.

Overall, the steps leading to the creation of a mesh suitable for FE analysis are labour-intensive. They typically rely on various segmentation techniques of the organ contour in 2-D cross-sections, and then on the reconstruction of the 3-D surface or volume e.g. by lofting and smoothing. Some techniques have been introduced to automate the reconstruction process, but most lack robustness, and important input is still required from the user.

In addition, most existing techniques go from medical images to 3-D modeling first (i.e. shape formation), and then from 3-D modeling to 3-D meshing (i.e. shape discretization), which is arguably a waste of time if one only needs a discretized shape as is typically the case in FE analysis. Unstructured triangular (or tetrahedral) meshes can be used to mesh nearly any 3-D surface (or volume, respectively), but these meshes are not always optimal in terms of computational speed or accuracy. Structured meshes based on quad and hexahedral elements have been shown to perform faster and with better accuracy. However, robust tools are currently missing to automatically generate structured meshes from 3-D medical images with minimum user input.

1.2 Motivation and objectives

Although FE analysis is a well-developed and mature method in the engineering world, mesh generation for complex geometries still may take several hours (De Santis et al., 2011). It is mainly because none of the traditional approaches is sufficient to effectively construct finite element meshes for arbitrarily complicated domains, with the consequence that a great deal of manual interaction is required in mesh generation (Zhang, 2013). This observation also holds true, of course, for human organs whose shape is usually quite complex. The situation is further compounded by the fact that segmentation from medical images is necessary before meshing can take place, with autonomous segmentation being one of the most difficult tasks in image processing (Gonzalez and Richard, 1993).

The main goal of the present work is to combine image segmentation and meshing efforts into one and thus produce a method to create FE meshes directly from 3-D medical images. In addition, the focus is placed on the creation of quadrilateral surface meshes and/or hexahedral volume meshes. This is because quadrilateral/hexahedral elements lend themselves to structured meshes that offer lower mesh complexity (simpler connectivity tables) and better solution quality than their triangular/tetrahedral counterparts. However, the generation of such meshes for arbitrary 3-D geometries is challenging as algorithms to generate meshes with provable guarantees on quality are available only for some restricted types of shapes.

Given the difficulties at stake, and while it will eventually be interesting to consider different organs, it was decided to focus first on the human aorta. Unlike the liver for instance, which has a complex 3-D volume, the aorta is, at its most basic, a

cylindrical tube following a curved path. This choice of focus allows for the difficulty of the problem to be incrementally raised from, say, 3-D surface meshing of the inside layer of the aorta, to the 3-D volume meshing of the blood volume contained in the aorta, and the 3-D volume meshing of the aortic wall with its branches.

1.3 Potential applications

With a robust tool to create patient-specific FE meshes directly from 3-D medical images of the aorta, engineering simulations could become more widespread to help clinicians determine the level of mechanical stress and strain in the tissues, which, for instance, is relevant to evaluating the risk of rupture in aortic aneurysms. Engineering simulations can also be used in the context of virtual surgery, where a surgical procedure to cut and replace a portion of the aorta by a synthetic graft can be simulated and its details optimized before it is actually carried out. These and related aspects will be discussed more fully later.

1.4 Framework and outline of this work

Chapter 1 presents the context and clarifies the motivations and objectives of the study. The potential applications are also briefly discussed in this chapter. Finally, the outline of the thesis is presented, along with the contributions of the work.

In Chapter 2, a literature review is made of relevant topics such as the anatomy and diseases of the aorta in so far as this knowledge is needed in the segmentation process. Segmentation methods are then discussed, with special emphasis on those

pertaining to cardiovascular structures using computed tomography (CT) and magnetic resonance (MR) images. This discussion is followed by features of structured as opposed to unstructured meshing methods. Existing techniques to generate patient-specific finite element meshes are also described. In short, Chapter 2 provides specific context to the thesis and identifies key concepts as well as the knowledge gap to be filled. In particular, the concept of 2-D gradient vector flow (GVF) is one that highlighted for its interesting potential toward image segmentation.

In Chapter 3, derivation of the 3-D GVF field is detailed. The properties of this field are illustrated using synthetic as well as real medical images.

Chapter 4 presents two failed initial attempts at using the 3-D GVF field towards our objectives. The lessons learned from these attempts helped shape the successful approach detailed in Chapter 5.

Chapter 5 proposes a novel methodology to automatically generate a structured surface mesh of the inner wall of the aorta. The methodology makes use of the 3-D GVF field for segmentation, and is very fast compared to existing methods. Illustration of the different steps in the methodology is made using a synthetic aortic dataset and three other real datasets. The chapter ends with the description of how the structured surface mesh information can be exported to a third-party FE software package.

Chapter 6 covers various methods to evaluate the accuracy of the reconstructed surface and the mesh quality produced by the methodology presented in Chapter 5.

Chapter 7 describes how the methodology used to obtain a structured surface mesh of the aortic inner wall can be expanded to generate hexahedral meshes for the aortic wall (solid domain) and/or the blood in the lumen (fluid domain).

Chapter 8 highlights ideas central to merging the disjointed structured meshes obtained for the aorta and its main branches in the aortic arch region. Due to time constraints, this chapter is more preliminary in nature than the previous ones. Along with Chapter 6 and 7, it is aimed at showing that the methodology proposed in Chapter 5 can indeed be expanded to meet all the objectives set out in Chapter 1.

Chapter 9 concludes the thesis by discussing the salient features and limitations of this work, as well as the recommended directions for its future extension.

1.5 Contributions

As will be shown, the methodology outlined in Chapter 5 is novel, effective and fast. It produces a structured surface mesh of the aorta with minimal user input, no a priori knowledge of the aorta's centerline, and no prior segmentation. As such, it is the major contribution of this work. Partial presentation of the methodology was given as an oral presentation at the 36th Canadian Medical and Biological Engineering Society conference (CMBEC 36/APIBQ 42, Ottawa, Ontario, Canada, May 21-24, 2013). The corresponding four-page article has been accepted for publication in a special conference proceedings issue of the Journal of Medical and Biological Engineering to appear in 2014. This journal is listed in Science Citation Index Expanded and Medline, among others. A poster presentation, after peer-review of an abstract, was also made at the American Society of

Mechanical Engineers 2013 Summer Bioengineering Conference, one of the leading international conferences in biomedical engineering (ASME SBC 2013, Sunriver, Oregon, USA, June 26-29).

A full-length manuscript is currently under preparation for publication in another peer-reviewed biomedical engineering journal.

A specialized MATLAB graphical-user interface (about 1,400 lines of code) was created to implement the ideas presented in Chapters 3, 5 and 7, making it possible for a user to load a medical imaging dataset, interactively select a region of interest in it, and compute its 3-D GVF field. Then, the user can interactively mesh the aortic wall and/or the fluid domain inside it and export the results to a third-party FE software package. The whole process takes on the order of 5 minutes from start to finish.

2 Literature review

2.1 Anatomy and common diseases of the aorta

The aorta is the human body's main artery. It commences from the outlet of the left ventricle of the heart, goes up (ascends), forms an arch with branches, then goes down (descends) through the chest (thorax) and through the belly (abdomen), where it branches off into two smaller arteries called the common iliac arteries (Kovacs, 2010). The aorta is traditionally divided into the ascending aorta, the aortic arch, and the descending aorta. The descending aorta is further subdivided into the thoracic aorta and the abdominal aorta (Figure 2.1)

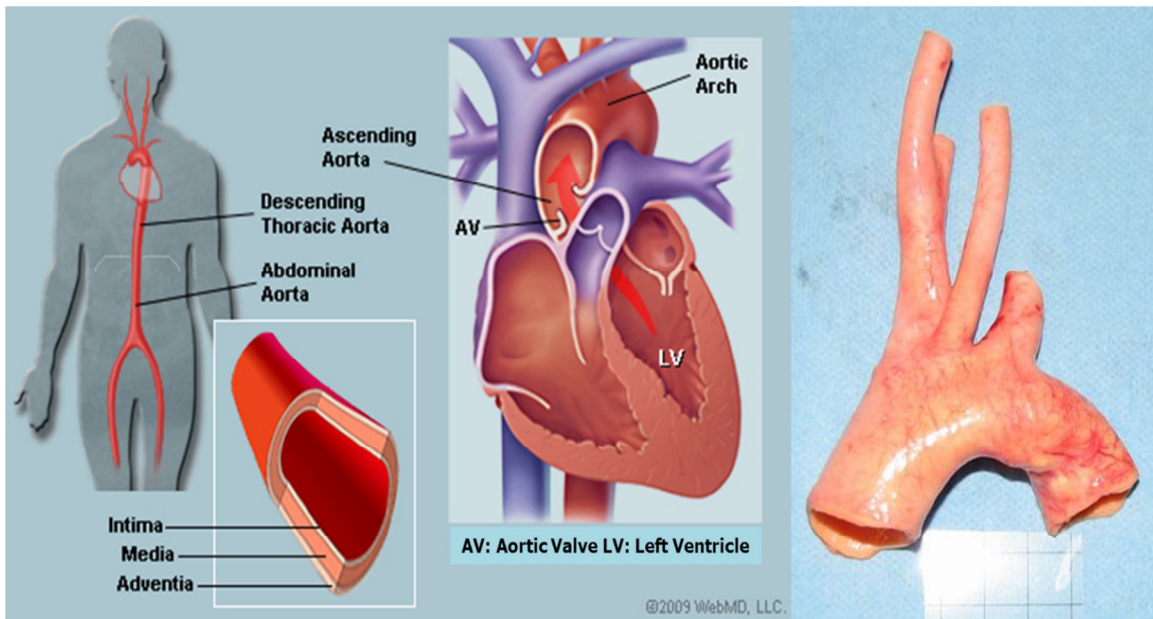


Figure 2.1: Left: the entire aorta and its major branches (WebMD, 2009) ; Middle: The heart, aortic arch and aortic valve (WebMD, 2013); Right: the aortic arch and its branches, with the brachiocephalic trunk (left), the left common coronary artery (middle) and left subclavian artery (right) (Photograph taken in Dr. Labrosse's lab of an excised human aorta).

A healthy aortic arch approximately forms a non-planar 180° sector of a torus, which can be estimated by giving its center and radius (Figure 2.2). Additionally, the radius of the aorta decreases linearly with the distance from the heart. Goldwin (Goldwin 1990) established the range of possible radial dimensions in healthy subjects in the ascending and descending aorta (Table 2.1).

Table 2.1: Normal Aortic Dimensions in adults (Goldwin 1990).

Location	Radius Size	Acquisition Technique
Ascending aorta	12.5-19 mm	CT
Descending aorta	8.5-14 mm	CT

Normal diameters for the thoracic aorta in adults was investigated in more detail by Hager et al. using helical computed tomography (Hager et al., 2002). The aortic diameters were measured at multiple levels: aortic valve sinus, ascending aorta, aorta proximal to innominate artery, proximal transverse aortic arch, distal transverse aortic arch, aortic isthmus, descending aorta at level of diaphragmatic wall of left ventricle.

With 3-D reconstruction of the aorta in mind, it is important to describe aortic diseases that can affect its shape and the structure of its wall. Aside from the normal general stiffening of the aorta that is thought to be a natural consequence of ageing, inherited diseases can lead to aortic enlargement (aneurysms) or tears and delamination (dissections). Complex thoracic aortic disease includes various types of acute and chronic

aortic problems, such as aortic root or ascending aortic dilation/aneurysm and aortic arch aneurysm, and acute aortic dissections. All aortic diseases are associated with microstructural changes, either to the content or architecture of the connective fibres elastin or collagen (Tsamis et al., 2013).

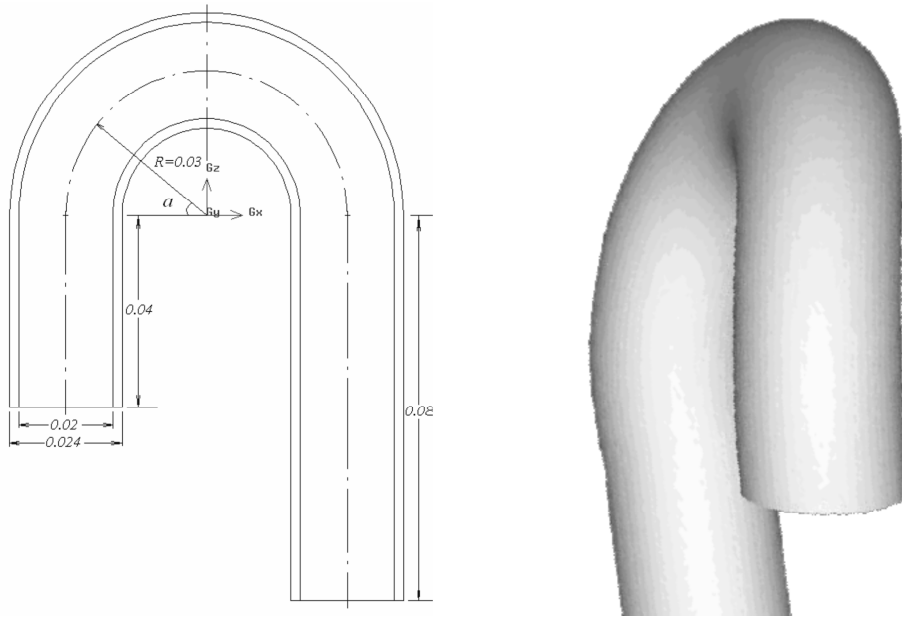


Figure 2.2: Left: Planar simplification of the aortic arch introduced by (Gao et al., 2005); Right: A 3-D non-planar model developed by (Fujioka et al., 2000).

2.1.1 Aortic aneurysms

An aneurysm is a weak point in a blood vessel wall, most commonly in an artery such as the aorta. Blood pressure tends to push the weakened section of the arterial wall outward, forming a permanent balloon-like projection. This condition by itself is not especially harmful, unless the artery is prone to rupture. The most common type of aneurysm is an abdominal aortic aneurysm (AAA). AAA is typically defined as a greater than 50% increase in the normal aortic diameter. It is a major source of morbidity and mortality for

those 65 years of age and older with approximately 200,000 people diagnosed annually in the United States alone. Aneurysms are typically slow to develop and are often asymptomatic prior to rupturing. Mortality increases dramatically after the rupture with only 10% survival to surgery (Hackam et al., 2006).

Accurate measurements of AAA geometric characteristics are critical for a reliable estimate of AAA rupture risk. The association between AAA diameter and 12-month AAA rupture risk is illustrated in Table 2.2.

Table 2.2: 12-month AAA rupture risk by diameter (Moll et al., 2011).

AAA Diameter	Rupture Risk (%)
30–39	0
40–49	1
50–59	1.0–11
60–69	10–22
>70	30–33

Recent studies have shown that aneurysm diameter is not the sole determinant for rupture, although it is the only one practically considered by clinicians. Peak wall stress has been shown to be a more reliable parameter than maximum diameter for predicting aneurysm rupture (Vorp, 2007; Auer & Gasser, 2010). Using CT images, Helderma et al. designed finite element models to calculate wall stress and aneurysm expansion rates in a large pool of patients (Helderma et al., 2010). Different methods of biomechanical analyses regarding aortic aneurysms and aortic dissection (as well as carotid and coronary

stenosis) are reviewed in (Ricotta et al., 2008) and (Martufi et al., 2009). In AAA, the possible presence of a thrombus (clot) poses major challenges for segmentation because of imaging limitations in differentiating the actual aortic wall from the clot. In addition, according to Auer and Gasser, current structural model development for AAA is still a cumbersome and time consuming process (Auer & Gasser, 2010).

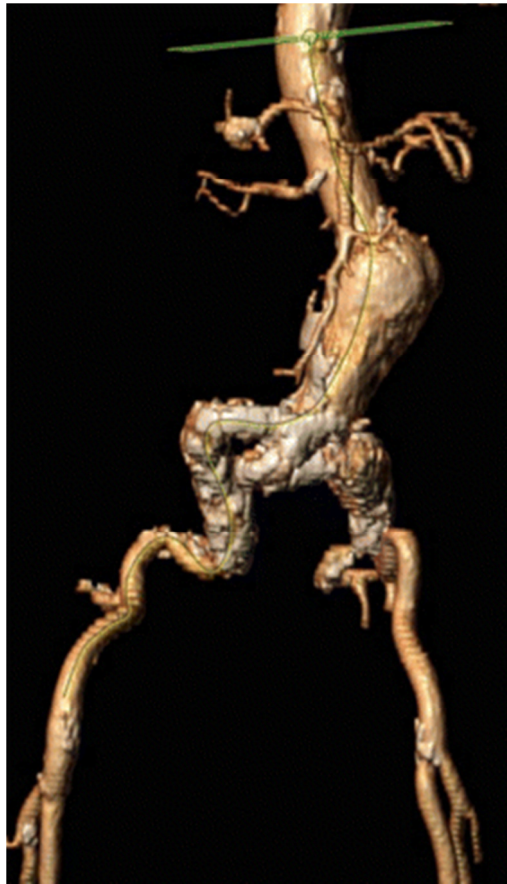


Figure 2.3: Preoperative computed tomography reconstruction of a 7.4 cm abdominal aortic aneurysm (Metcalf et al., 2011).

2.2 Segmentation methods for the cardiovascular system

Medical images can be defined as 2-D or 3-D arrays of gray scale intensities provided by medical scanners. The information is usually stored using the Digital Imaging and Communications in Medicine (DICOM) standard. The segmentation process consists in delineating and extracting the geometry of interest (Antiga et al., 2008). Segmentation methods are very numerous and varied; there are also many different approaches to classifying them. In the work of Withey and Koles, segmentation methods are divided into three generations depending on their level of complexity in image analysis: first generation methods include the simplest methods of segmentation i.e. based on image intensity; second generation methods utilize uncertainty models and optimization techniques, and third generation methods combine first and second generations (Withey & Koles, 2007). Quite differently, in the research of Pham et al, segmentation techniques are divided into eight groups, namely thresholding, region growing, classifiers, clustering, Markov random field, artificial neural network, deformable models and atlas guided methods (Pham et al., 2000). Yet differently, in Radaelli and Peiro's work, segmentation methods are grouped into intensity-based and gradient-based, model-based and region-based algorithms (Radaelli & Peiro, 2010). In medical imaging applications, multiple techniques are often used in conjunction with each other for solving segmentation problems.

Therefore, in what follows, segmentation methods focus on segmentation of the aorta only, and are categorized according to their goals. With image-based methods, the goal is to segment the images and obtain an unspecified shape; with model-based

methods, the goal is to transform an arbitrary shape to match an organ contained in the images.

2.2.1 Image-based methods

2.2.1.1 Thresholding methods

Thresholding is one of the simplest methods used for image segmentation. It is often used as an initial step in a sequence of image-processing operations. In this method, regions with different intensity levels are distinguished using thresholds. As illustrated in Figure 2.4 in the case of a single-threshold (T), a thresholded image $G(x, y)$ may be defined as

$$G(x, y) = \begin{cases} 1 & f(x, y) > T \\ 0 & f(x, y) < T \end{cases}$$

Threshold value selection can be done by the user in an interactive manner, or based on a statistical measure (e.g. the mean of the image intensity). Thresholding is often performed iteratively, based on the operator's visual assessment of the resulting segmentation. There are certain limitations to thresholding. In its simplest form, it generates only two classes and since it does not take into account the spatial characteristics of an image, thresholding cannot be applied to multichannel images that are built from several monochrome intensity arrays that each correspond to one particular wavelength range. Moreover, thresholding is sensitive to noise and intensity heterogeneity (Pham et al. 2000).

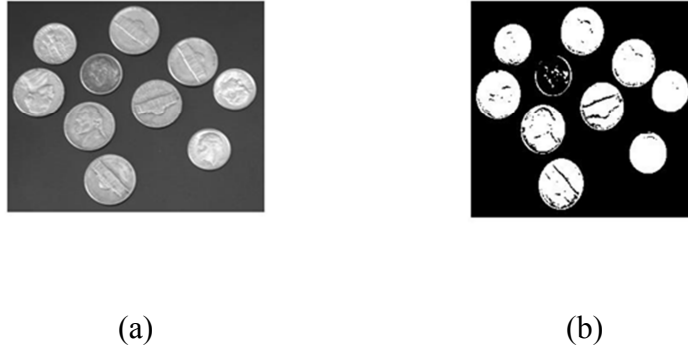


Figure 2.4: Thresholding a grayscale image: (a) Original image, (b) Thresholded image.

2.2.1.2 Region growing method

In the region growing segmentation method, a seed point is manually chosen by the operator, and all pixels connected to the initial seed based on some predefined criterion, such as image intensity, are extracted. When the criterion is applied continually, the region grows until an edge in the image is met and the volume of the object is defined by identification of similar, connected pixels. The primary disadvantage of region growing is that it requires manual intervention in placement of the seed point in the image. It is sensitive to noise and it may form holes and disconnected regions in presence of noise (Antiga et al., 2002). On the other hand, different regions can get connected (Pham et al., 2000). The other disadvantage is that different choices of seeds may give different segmentation results. Region growing, similarly to thresholding, is usually employed with other segmentation methods.

Macia et al. used a 3-D region growing method for lumen segmentation. The lumen of the aorta is the opening through which blood flows. In their method the selection of two or more seed points is manual. The thrombus contour is modeled as a radial function and the internal and external radius of the thrombus of the aneurysm are in

cylindrical coordinates. The function for the external radius is obtained by local and slice-level analysis of connected components and an a priori knowledge of the location, size and intensity of the thrombus. Although the segmentation speed achieved by their method is good, the parameters must be manually tuned for a large number of datasets (Macia et al., 2009).

Pohle and Toennies have worked on a fully automatic region growing algorithm that learns its homogeneity criterion automatically from characteristics of the region to be segmented (Pohle & Toennies, 2001). The method is based on a model that describes homogeneity and simple shape properties of the region. Parameters of the homogeneity criterion are estimated from sample locations in the region. These locations are selected sequentially in a random walk starting at the seed point, and the homogeneity criterion is updated continuously. The methods were tested by segmenting the inner aortic wall in abdominal aortic aneurysms, among other structures, in CT and MR images. Figure 2.5 shows a representation of region growing method. The selected seed point and the growing process is shown.

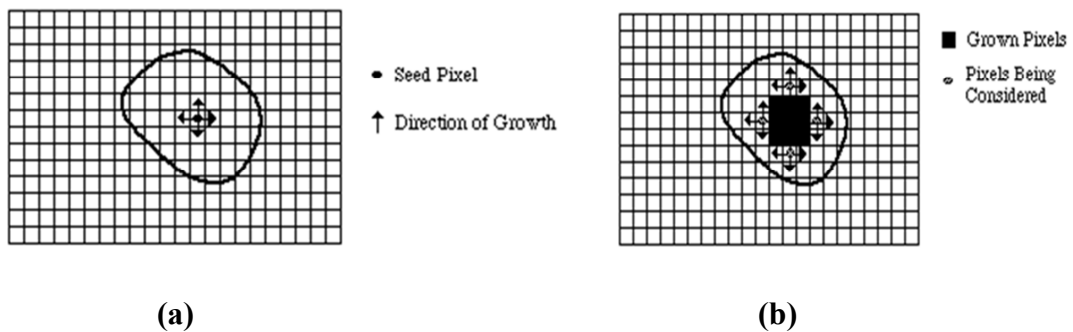


Figure 2.5: Region Growing, (a): Start of growing region, (b): Growing process after iteration (Marshall, 2008).

2.2.1.3 Classifier methods

Classifier methods are known as supervised methods because they require training data that are manually segmented and then used as references for automatically segmenting new data. There are a number of ways in which training data can be applied in classifier methods. A simple classifier method is the nearest-neighbour classifier method, where each pixel or voxel is classified in the same class as the training datum with the closest intensity. The k-nearest-neighbour (kNN) classifier method is a generalization of this approach, where the pixel is classified according to the majority vote of the k-closest training data (Pham et al., 2000). In (Olabarriaga et al., 2004), abdominal aortic aneurysms from CT images were segmented using non-parametric pattern classification technique. Classifier methods are relatively efficient due to their non-iterative nature. In addition, unlike thresholding methods, they can be applied to multichannel images. The main weakness of classifier methods is that they do not generally perform any spatial modeling. The other disadvantage of classifiers is the requirement of manual interaction to obtain training data.

2.2.1.4 Clustering methods

Clustering methods use the same algorithm as classifier methods but without training the data. These methods implement an unsupervised classification of data into regions of homogeneous characteristics. Clustering algorithms iteratively train themselves using available data. They alternate between segmenting data and characterization of the property of each class. Clustering can be categorized into hard and soft clustering. In hard clustering, membership of a pixel belongs to one class. In soft clustering a pixel is categorized into a class depending on the probability of belonging to that class. Three

commonly used clustering algorithms are: k-means, the fuzzy-c means and the expectation-maximization algorithm. In another work, stent grafts in abdominal aorta were segmented using in part clustering algorithms (Klein et al., 2008). Although clustering does not need data training, it needs initial segmentation. Like classifier methods, clustering algorithms do not directly incorporate spatial modeling and are sensitive to noise and intensity inhomogeneities (Withey & Koles, 2007).

2.2.1.5 Artificial neural networks

An artificial neural network (ANN) is a parallel network of processing elements or nodes that simulate biological learning. Each node in an ANN is capable of performing elementary computation. During training of the system, weights are associated with each node automatically. Depending on the weights and the computational result of each node as given input, the image produces an output i.e. the segmented image. ANNs are mostly used in medical imaging to implement classifier methods, where the weights are determined by training data and the ANNs are used to segment the new data. The ability of ANNs to naturally incorporate spatial information regarding the structures and tissue types of interest is expected to provide significant increases in reliability and robustness compared to solely statistical methods. Katz and Merickel illustrated how ANNs can be used to classify and segment multispectral images (a multispectral image is one that captures image data at specific frequencies across the electromagnetic spectrum, as typically acquired by remote sensing radiometers) to distinguish the aorta from surrounding tissues in MRI images (Katz & Merickel, 1990). Three-dimensional reconstruction of abdominal aortic aneurysms was performed using an ANN by De Biasi et al. as well (De Biasi et al., 2002). ANNs can also be used to implement clustering

methods. Although ANNs are inherently parallel, their processing is usually simulated on a standard serial computer which reduces the computational advantage of this technique.

2.2.1.6 Markov random field models (MRF) / Statistical pattern recognition

The Markov random field (MRF) model is a statistical tool that can be used with other segmentation techniques toward robust segmentation. MRF models represent spatial relation between neighbouring pixels. Given that in medical images most pixels belong to the same class as their nearby pixels, this method is often used. The local correlation value implies that it is almost impossible to have regions with small numbers of pixels. MRF models are widely used to model the inhomogeneities that occur in MR images and digital mammograms. Rueckert et al. presented an algorithm for tracking the aorta in cardiovascular MR images (Rueckert et al., 1997). Their algorithm first estimates the location and diameter of the aorta and then refines this estimate using the MRF model. Other people have used implicit contours in Markovian method to segment MR images (Jodoin et al., 2008). The proper selection of the parameter which controls the spatial interactions is an important factor in MRF. This could be considered as one of the drawbacks of this technique. On the other hand, MRFs need computationally intensive algorithms.

2.2.1.7 Edge tracing

In the edge tracing segmentation technique, edge pixels with adjacent neighbour connectivity are followed sequentially and heuristically and all gathered in a list to represent an image boundary. Then, an edge image is produced by edge detection.

2.2.2 Model-based methods:

Image-based methods as presented above are interesting for the concepts they use and illustrate, however cannot be used directly toward our objectives, because they do not provide any information about a mesh that could be used to describe the segmented geometry. Therefore, model-based methods are now presented, as they conceptually get closer to having this desirable feature.

2.2.2.1 Atlas-guided segmentation

Atlas-guided medical image segmentation is a powerful tool for segmenting anatomical structures when an atlas of the structures is available. The atlas is generated by compiling information about the anatomy that requires segmentation. Like classifiers, this atlas is used as a reference frame for segmentation. A one-to-one transformation from the pre-segmented image to atlas image is necessary to initiate the segmentation. This process is known as atlas wrapping. Atlas wrapping can be done through linear or non-linear transformations. Atlas guided segmentation is particularly helpful in steady structures like the brain. However, with dynamic structures like the heart, additional computation is necessary for segmentation. Implementation and preliminary results of a technique for creating a comprehensive probabilistic atlas of the human brain based on high-dimensional vector field transformations is shown in (Thompson & Toga, 1997). Many atlases contain probabilistic information. In many cases, the use of an atlas requires distinct steps for registration and segmentation. Image registration is the process of overlaying two or more images of the same scene taken at different times, from different viewpoints, and/or by different sensors. It geometrically aligns two images—the reference and sensed images (Zitová & Flusser, 2003).

According to Isgum et al., there are three different atlas based segmentation methods: 1) single atlas-based segmentation, 2) average-shape atlas-based segmentation, and 3) multi-atlas-based segmentation (Isgum et al., 2009).

Depa et al. demonstrated accurate automatic segmentation that is robust to the high anatomical variations in the shape of the left atrium in a clinical dataset of MRA images (Depa et al., 2010). Zhang et al also present an atlas-based algorithm for constructing three-dimensional cubic finite element meshes of the whole human heart from tomographic patient image data (Zhang et al., 2012).

2.2.2.2 Deformable models

Deformable models are used for boundary extraction and tracking in 2-D and 3-D images. The most widely used deformable models include active contours, also known as snakes or deformable contours (Kass et al., 1988) and active surfaces or deformable surfaces (Li & Acton, 2007). In 2-D, active contours use closed parametric curves deforming under the influence of internal and external forces to delineate region boundaries. The curve has to be initialized manually, typically close to the region of interest. The internal forces are used to preserve the smoothness of the curve and the external forces are computed from the image to drive the curve towards the region of interest.

Deformable model methods can be divided into two classes (Kirbas & Quek, 2003): parametric active contour models (Kass & Witkin, 1988; Xu & Prince, 1998) and geometric active contour models (Osher, 1988).

Parametric active contour models minimize a defined energy function, and the contour in an image is the position at which the energy is minimized. An initial contour

that is arbitrarily defined by a user is deformed to the object contour through the minimization of energy. However, since this method does not allow adaptation to topological changes (e.g. a bifurcation), controlling the topological changes is not easy. On the other hand, geometric active contour models that use the level set theory can detect separating and merging contours (Kim et al., 2007).

Deformable models have been widely used in medical image segmentation. Bro-Nielsen and Cotin used deformable models for real time surgery simulation (Bro-Nielsen & Cotin, 1996). While Park et al presented a method for analyzing the motion of the heart's left ventricle from tagged magnetic resonance imaging (MRI) data based on the development of a class of physics-based deformable models (Park et al., 1996). Using a generalization of such models, Sarry and Boire tracked the coronary arteries from biplane angiographic sequences of parametrically deformable models (Sarry & Boire, 2001). Deformable models were reviewed by McInerney and Terzopoulos (McInerney & Terzopoulos, 1996).

As shown by Antiga et al., the constraint to maintain a smooth contour makes deformable models more robust against noise than many other techniques (Antiga et al., 2002). Shape prior information is easily incorporated in deformable models and prevents the active contour model from getting trapped into false edges. This leads to what is known as active shape models, which are modified from deformable models with the intention of limiting the extent of model deformation. In this approach, a set of landmark points on the object boundary are statistically represented and variation of each landmark point across a set of training images is analysed (Withey & Koles, 2007). Therefore, active shape models combine statistical knowledge of object shape and shape variation

with local appearance near object contours. Further improvements have been proposed (Sollie, 2002; de Bruijne et al., 2002). However, the method may require extensive manual initialization, as well as intervention during processing.

Biesdorf et al used the benefits from the robustness of model-based segmentation and the accuracy of elastic registration (Biesdorf et al., 2012) (Figure. 2.6). They combined 3-D model-based segmentation with elastic image registration to integrate both segmentation and registration scheme for the quantification of the aortic arch morphology from 3-D tomographic images. Their joint approach is based on an energy-minimizing functional. In comparison to a pure model fitting approach (e.g. Wörz et al., 2010), the joint approach can cope with a larger spectrum of vessel shapes, in particular, with shapes that deviate significantly from the underlying model used for segmentation (e.g., in pathologies such as aneurysms).

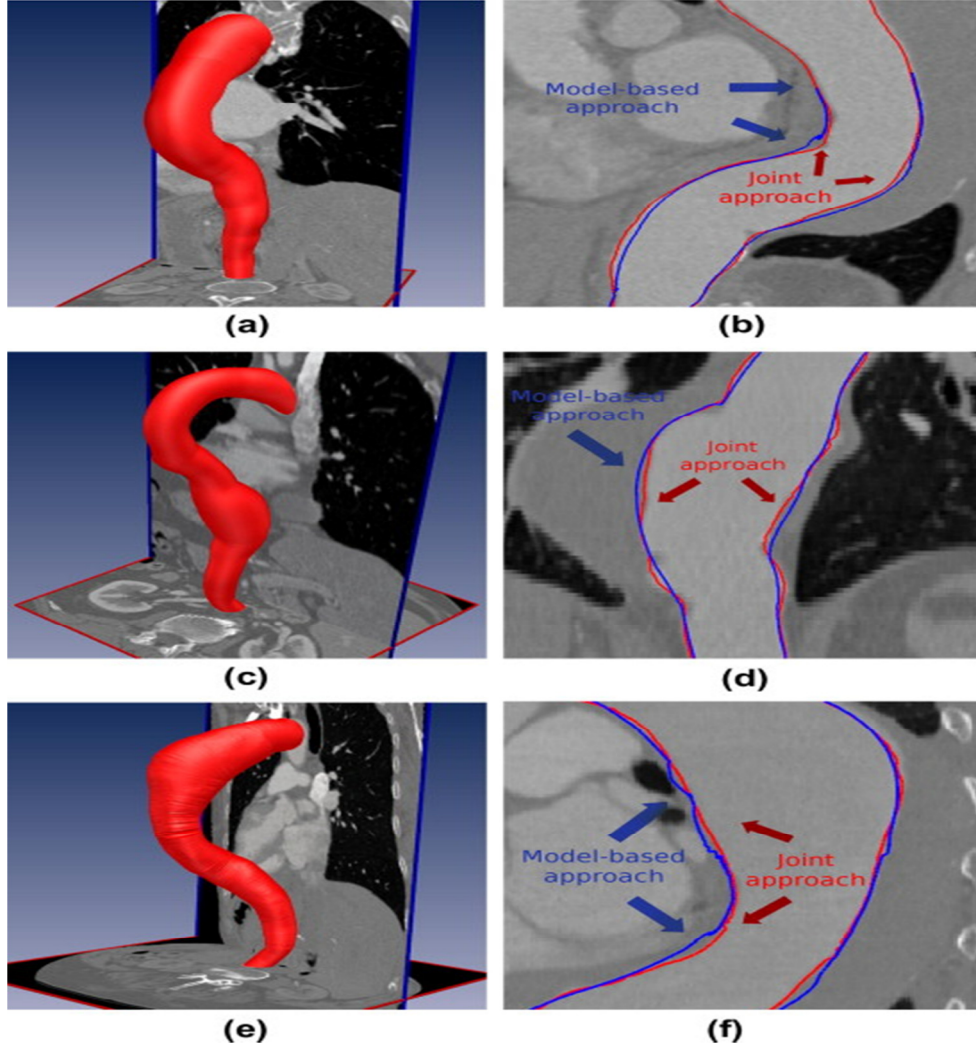


Figure 2.6: Joint approach: (a), (c), (e) Segmentation results of the 3-D joint approach for three 3-D CTA images showing a pathology. (b), (d), (f) Vessel contours using the model-based approach (blue) and the 3-D joint approach (red) for a section of the 3-D CTA images (Biesdorf et al., 2012).

There are different methods to represent model deformation: in the Lagrangian method, material points are followed during the evolution; in the Eulerian method, the motion of the model is observed through the evolution of scalar field values at fixed points in space. The first technique is commonly used in parametric active contours. The second technique is part of a general formulation of the evolution of implicit deformable models known as level set approach (Radaell & Peiro, 2010). The classical level set

formulation was introduced in Malladi, 1995. It uses an implicit representation of curves in the form of a partial differential equation to track boundaries, without geometrical assumptions or a-priori shape knowledge.

Loncaric et al. developed a 3-D deformable model based on the level set approach to generate a 3-D model of AAA from which accurate measurements could be obtained. Their method provides easy segmentation for complex geometries, including branches (Loncaric et al., 2000).

Kurugol et al. used a 3-D level set approach in non-contrast CT. Instead of refining the centerline position after initialization, as usually done, in this work, they refined the initial aorta surface position using a level set algorithm based on standard gradient and smoothness terms and a distance regularization term (Chan & Vese, 2001), (Chunming, Li 2005) to restore the regularity of evolving level set function (Kurugol et al., 2012). For some regions, where edges are partially missing, 3-D smoothness constraints stop leakage locally. This method works fine as long as prior information about the boundary surface exists.

Using the level set method Zohios et al. were able to segment the aortic thrombus and outer wall boundaries as well as aortic calcifications (Zohios et al., 2012). Figure 2.7 shows an example of applying 3-D deformable model to CT images of the aorta using the level set algorithm (Loncaric et al., 2000)

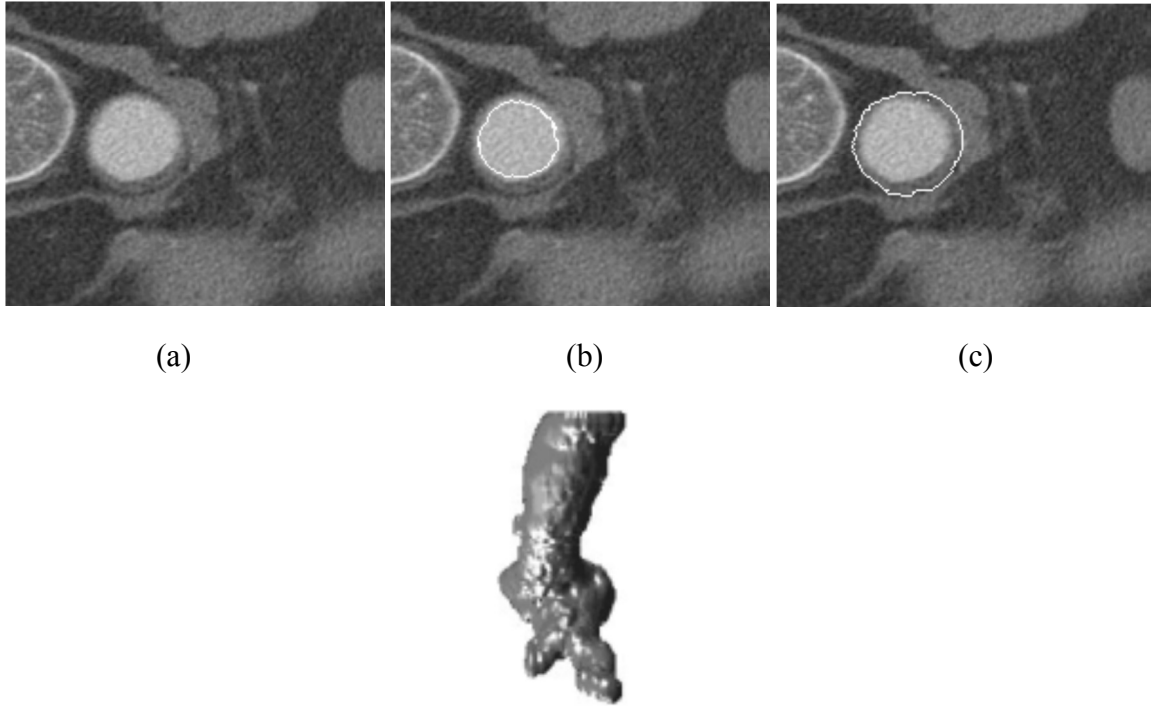


Figure 2.7: Resulting 3-D model of abdominal aorta (Loncaric et al., 2000)

(a): Slice of CT images, (b): Aorta inner boundary, (c): Aorta outer boundary.

Zhao et al. segmented the aorta in MRI images. They used an automated segmentation method combining the level-set approach with an optimization process to delineate the aortic surface (Zhao et al., 2006). To improve the segmentation performance, a 3-D vessel enhancement filter was first applied to the original images to suppress noise and image background for tubular objects (Frangi et al., 1998).

Galante et al. modified the classical level set approach to segment the aorta in dynamics (Galante et al., 2007).

Overall, the level set method is capable of handling different image dimensionalities and topological changes. This method has been shown to be powerful approach for segmenting both the lumen and thrombus. However, in spite of all its interesting features, the implicit nature of the contour being reconstructed in the level set

approach does not lend itself well to the creation of a structured mesh as is the objective of our work.

Although none of the deformable models presented above were found to match the specific requirements of our project, it is interesting to look closely at the mathematical development of parametric active contours, for it has progressively led the research community to the formulation of the gradient vector flow which has turned out to be a core concept in the present work.

2.2.3 Mathematical development of parametric active contours

Unlike passive models that do not interact with each other or with external forces, active contour models react to applied forces (such as gravity), to constraints (linkages), to ambient media or to obstacles (McInerney & Terzopoulos, 1996). Active contours are well suited for segmentation of non-rigid boundaries which are frequently found in biomedical images. Active contour models involve the solution of an energy optimization problem to minimize the sum of internal and external energies. Minimization of internal energy ensures preservation of smooth contour and minimization of external energy term ensures fitting of the active contour to a desired location. To reach this goal, there are algorithms which deform a pre-defined contour to lock onto features of interest within an image, such as edges. Thus, the minima of the chosen potential energy coincide with the interesting features of the image however the initial position must be provided for active contours. They can theoretically work in 2-D, 3-D or in even higher dimensions. The following presentation is for 2-D problems.

From a mathematical perspective, an active contour is a 2-D parametric curve represented by $\mathbf{X}(s) = \{x(s), y(s)\}, (0 \leq s \leq 1)$ that deforms within the image domain

so as to minimize the following energy functional (Kass et al., 1988; Loncaric et al., 2000):

$$E(\mathbf{X}(s)) = E_{int}(\mathbf{X}(s)) + E_{ext}(\mathbf{X}(s)) \quad (2.1)$$

where E_{int} indicates the internal deformation energy as:

$$E_{int}(\mathbf{X}(s)) = \int_0^1 \left(\omega_1 \left| \frac{d\mathbf{X}(s)}{ds} \right|^2 + \omega_2 \left| \frac{d^2\mathbf{X}(s)}{ds^2} \right|^2 \right) ds. \quad (2.2)$$

ω_1, ω_2 are weighting parameter functions, typically constants, that control the elasticity by producing stiffness through tension and bending, respectively. E_{ext} Indicates the external energy determined as:

$$E_{ext}(\mathbf{X}(s)) = \int_0^1 \omega_3 f(\mathbf{X}(s)) ds. \quad (2.3)$$

Classically, the external energy term ensures fitting of the active contour to a desired location, e.g. the edge. Depending on the application, the definition of the external energy can be different. Typical external energies for gray-level image $I(x, y)$ for seeking edges are given as:

$$E_{ext}(\mathbf{X}(s)) = -(\nabla I(x, y))^2 \text{ or}$$

$$E_{ext}(\mathbf{X}(s)) = -|\nabla[G_0(x, y) * I(x, y)]|^2$$

where $G_0(x, y)$ is a 2-D Gaussian function with standard deviation σ . $*$ denotes linear convolution, and ∇ denotes the gradient operator.

The active contour corresponds to a minimum of the energy functional $E(\mathbf{X}(s))$ in (2.1). To minimize the total energy of the active contour, conditions $E'(\mathbf{X}(s)) > 0$, and $E''(\mathbf{X}(s)) = 0$ must be satisfied. $E(\mathbf{X}(s))$ is a functional, therefore variational calculus is required to obtain the minimum of the energy function. At the minima of (2.1), the contour must satisfy the Euler-Lagrange equation:

$$\omega_1 \mathbf{X}(s)'' - \omega_2 \mathbf{X}(s)'''' - \nabla E_{ext}(\mathbf{X}(s)) = 0. \quad (2.4)$$

This equation can be viewed as a force balance between:

$$\mathbf{F}_{int}(\mathbf{X}(s)) + \mathbf{F}_{ext}(\mathbf{X}(s)) = \mathbf{0}, \quad (2.5)$$

where:

$$\mathbf{F}_{int}(\mathbf{X}(s)) = -E_{int} = \omega_1 \mathbf{X}(s)'' - \omega_2 \mathbf{X}(s)'''' , \quad (2.6)$$

$$\mathbf{F}_{ext}(\mathbf{X}(s)) = -E_{ext}(\mathbf{X}(s)) .$$

To solve (2.5), $\mathbf{X}(s)$ is classically treated as a function of time t . The solution is obtained by reaching the steady state solution of the following gradient descent equation (Li & Acton, 2007):

$$\frac{\partial \mathbf{X}(s)}{\partial t} = \omega_1 \mathbf{X}''(s) - \omega_2 \mathbf{X}''''(s) + \mathbf{F}_{ext}(\mathbf{X}(s)) \quad (2.7)$$

starting from an initial contour $\mathbf{X}(s, 0)$.

A numerical solution to (2.7) on a discrete grid can be achieved using a finite difference approach (Tiilikainen, 2007). The continuous contour $\mathbf{X}(s)$, $s \in \{0,1\}$ is sampled and represented by a set of m discrete points $\mathbf{x}_i, i \in \{0,1,\dots,m-1\}$. These sample points are referred to as active contour elements. The update procedure for the entire contour can be written in matrix form as:

$$(\mathbf{I} + \tau \mathbf{A}) \mathbf{X}^{t+1} = \mathbf{X}^t + \tau \mathbf{F}^t, \quad (2.8)$$

where $\mathbf{I}$ is the $m \times m$ identity matrix,

$$\mathbf{X}^t = \{\mathbf{x}_0^t, \mathbf{x}_1^t, \dots, \mathbf{x}_{m-1}^t\}^T \text{ and}$$

$\mathbf{F}^t = \{F_{ext}(\mathbf{x}_0^t), F_{ext}(\mathbf{x}_1^t), \dots, F_{ext}(\mathbf{x}_{m-1}^t)\}^T$ are $m \times 2$ matrices representing the positions and the external forces of active contour elements at time t , respectively.

$\mathbf{A}$ is an $m \times m$ matrix used to compute the internal force and τ is the time step size (Cohen & Cohen, 1993).

As shown above, instead of an energy minimization problem, the solution of the deformable model is formulated as a force balance equation; therefore the choice of $\mathbf{F}_{ext}$ in (2.7) has a profound impact on both the implementation and the behaviour of the deformable model. In the following sub-sections, examples of external forces are discussed along with their pros and cons.

2.2.3.1 Gradient potential force

The classical approach of Kass et al. uses squared image gradient magnitude as the basis of active contour external force $\mathbf{F}_{ext} = -\nabla I^2$. External energy computed from image gradient is sensitive to noise and may detect false edges. Moreover, initialization of an active contour in a homogeneous region fails to converge (Figure 2.8).

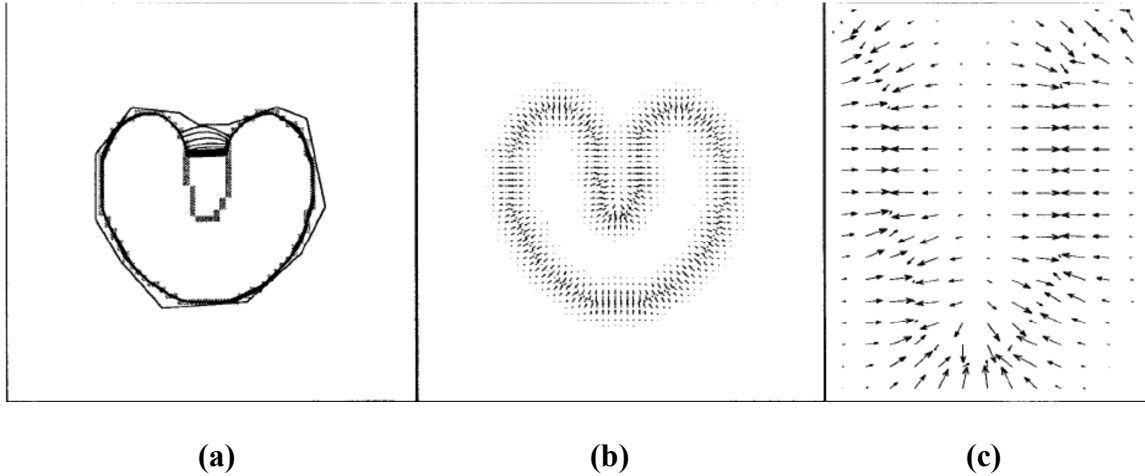


Figure 2.8: (a) Convergence of active contour using (b) gradient potential force, (c) close-up within the boundary concavity (Xu & Prince, 1998).

Since the image gradient is zero in homogeneous regions, no edge force would act on the active contour to push it towards the desired edges. Hence, active contour models based on gradient magnitude as external energy are initialization dependent and have a

limited capture range (Borgefors, 1984). To solve this problem, different energy potentials as external forces have been developed over the years.

2.2.3.2 *Distance potential force*

The distance potential force proposed by (Cohen & Cohen, 1993) is one solution to the limitation of gradient potential force in case of binary images. This approach derives an external force by computing a signed distance at each point on the deformable contour. This signed contour is calculated by determining the closest boundary point or other image features.

The distance map $D(x, y)$ for a binary image $I(x, y)$ is computed as

$$D(x, y) = \min_{(p, q) \in \{(a, b): I(a, b)=1\}} [d(x, y; p, q)] \quad (2.9)$$

where $d(x, y; p, q)$ is a distance metric between the boundary points and deformable contour and can be the Euclidean distance as:

$$D(x, y; p, q) = \sqrt{(x - p)^2 + (y - q)^2}. \quad (2.10)$$

Once $D(x, y)$ is computed, the distance potential field, used as an external force vector is given by:

$$\mathbf{P} = -\mathbf{D}(-I); \mathbf{D}(I) = (-D_x(x, y), -D_y(x, y)) \quad (2.11)$$

which governs the active contour.

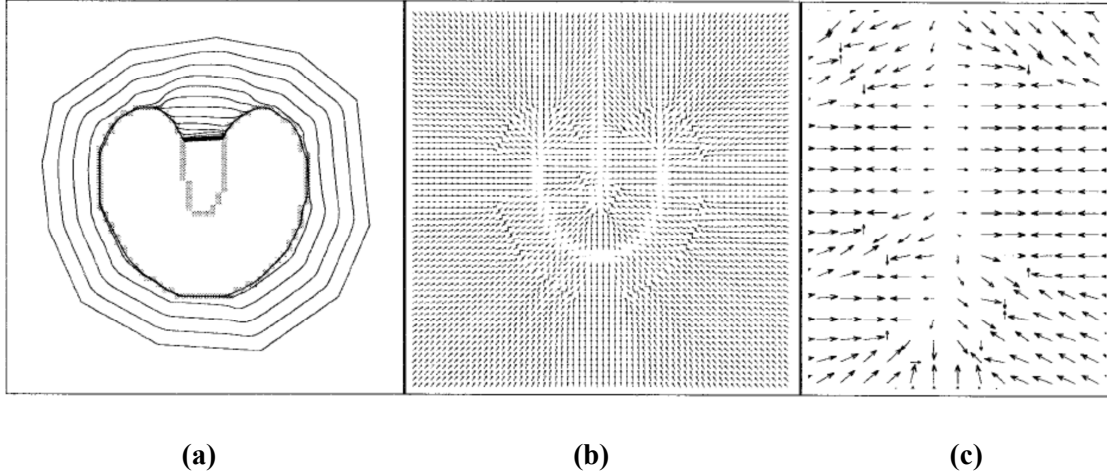


Figure 2.9: (a) Convergence of active contour using (b) Distance potential force, (c) close-up within the boundary concavity (Xu & Prince, 1998).

As shown in Figure 2.9 the distance potential forces have a larger capture range compared to gradient potential forces. However the range is still limited and the active contour requires initializing close to the desired contour. To solve this problem an alternative is the balloon force as discussed next.

2.2.3.3 Pressure force

The pressure force used in balloon models is a force that pushes the active contour either outward (inflation) or inward (deflation) (Cohen, 1991). The pressure is normal to the active contour. (Cohen & Cohen, 1993) used a finite element formulation and balloon active contours to solve the problems with the original active contour model of Kass et al. They also showed a 3-D generalization of the balloon active contour as a 3-D deformable surface. (McInerney & Terzopoulos, 1996) used pressure active contours to segment cardiac 4-D images. Although the pressure force can avoid the detection of false edges, and can push an active contour into boundary concavities, but cannot be too strong or “weak” edges will be overwhelmed (Xu & Prince, 1998). Another limitation of using

pressure force for active contours is that it must be initialized either inside or outside of the target object (Li & Acton, 2007).

2.2.3.4 *Gradient vector flow force*

If the object to be segmented has a concavity, none of the previously proposed models could force the active contour into the concavity. Although the image forces point correctly toward the object boundary, the range of the force field is small. Therefore, the curve is “pulled apart” but not made to progress into the concavity. A new force field was proposed by Xu and Prince to solve this issue (Xu & Prince, 1998). The underlying mathematical premise for this new force comes from the Helmholtz theorem, which states that the most general static vector field can be decomposed into irrotational and solenoidal components. They designed a new force field that has both the desired properties of a large capture range and the presence of forces that point into boundary concavities. An external force field $\mathbf{G}(x, y) = (u(x, y); v(x, y))$, called gradient vector flow (GVF) field, is constructed by diffusing the edge force away from the edges to homogeneous regions while at the same time keeping the constructed field as close as possible to the edge force near the edges.

In other words, $\mathbf{G}(x, y)$ is the vector field that minimizes the energy functional:

$$E_{GVF} = \frac{1}{2} \iint \mu \nabla^2 V + \nabla f^2 |V - \nabla f|^2 dx. \quad (2.12)$$

More specifically in 2-D, $E_{GVF} =$

$$\frac{1}{2} \iint \mu (u_x^2 + u_y^2 + v_x^2 + v_y^2) + (f_x^2 + f_y^2) ((u - f_x)^2 + (v - f_y)^2) dx dy \quad (2.13)$$

where μ is the non-negative parameter expressing the degree of smoothness of the field and $f(x, y)$ is the edge map derived from the image. Different incarnations of the edge

map will be discussed in the next chapter. The first integrand keeps the field smooth. It is similar to the classical Laplace's equation. The second integrand forces the vector field to resemble the initial edge force near the edge (i.e. where the edge force is high). The desired vector field is calculated from the energy functional using the calculus of variation, leading to:

$$\mu \nabla^2 u - (f_x^2 + f_{xy}^2)(u - f_x) = 0 \quad (2.14)$$

$$\mu \nabla^2 v - (f_x^2 + f_{xy}^2)(v - f_y) = 0 \quad (2.15)$$

Solving the above equations for (u, v) results in a GVF field that can be used to act as an external force in an active contour problem. A numerical implementation for finding (u, v) has been described by (Xu & Prince, 1998).

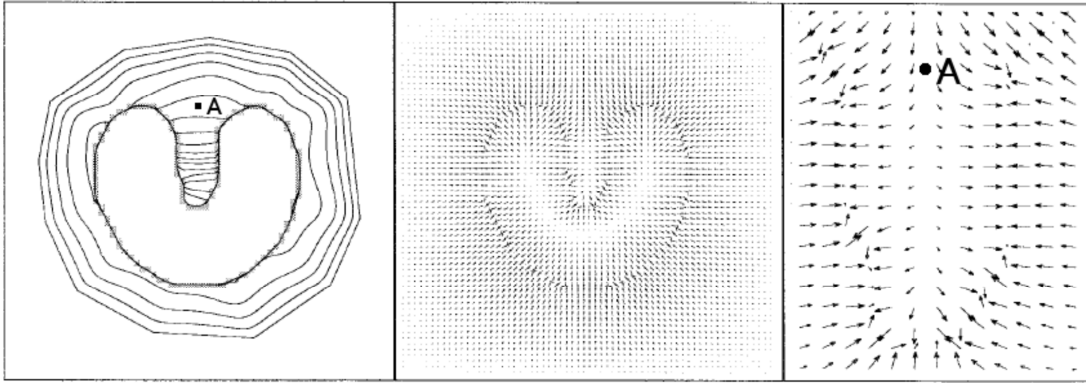


Figure 2.10: Gradient vector flow, Left: deformation of snake with GVF forces. Middle: GVF external forces. Right: close-up within the boundary concavity (Xu & Prince, 1998).

There are still some unsolved problems about GVF, such as the undefined relationship between the capture range and the parameters, noise sensitivity (especially impulse noise), and processing costs. However, it is retained as a powerful tool to transform a medical image into a vector field where individual vectors point toward the

boundaries of objects in the image and implicitly help in their segmentation. This is a promising feature for the objectives of the present work.

2.3 Mesh generation techniques

As mentioned previously, a mesh is a discretization of a continuous domain into simple shapes, such as triangles or quadrilaterals in 2-D and tetrahedra or hexahedra in 3-D. The elements of a mesh should satisfy certain criteria such as angles, conformity, and non-zero surface or volume (Gallier, 2006). These aspects will be discussed further in a later chapter. Meshes can be distinguished between structured and unstructured ones, as detailed below.

2.3.1 Structured mesh

A structured mesh (Figure 2.11) is characterized by regular connectivity that can be expressed as a two or three dimensional array following a uniform pattern. This restricts the element choices to quadrilaterals in 2-D or hexahedra in 3-D. Structured meshes have the property that, given a node in the domain, the location of its neighboring nodes is known by a simple relationship. That is, if the node has an index value for each of the coordinate directions defining the domain such as an (i, j, k) notation, where, for example, i ranges from 1 to N , the number of nodes in the x -direction, then the next node in the positive x –direction is known as the $i + 1^{\text{st}}$ node. Likewise for the j - and k -directions. Typically, the domain is discretized into cells (line segments in one dimension, areas in 2-D, and volumes in 3-D) by connecting mid-points (or other points

on lines connecting adjacent nodes) and building cell faces (or edges) from those connections (Hansen et al., 2005).

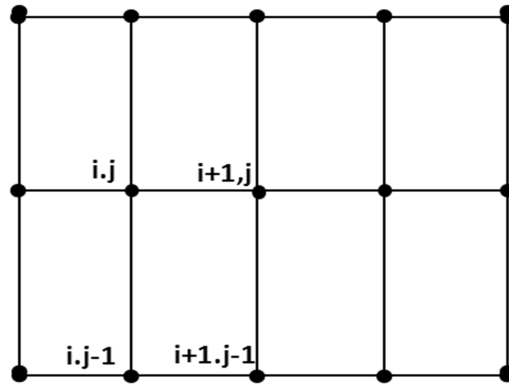


Figure 2.11: Structured mesh schematic.

Structured meshes come in three basic types: H-grids, O-grids, and C-grids. A grid of the O type is generated as a solid cube or rectangle in the computational domain and transformed from the physical domain, as presented in Figure 2.11. This type of numerical grid has only two boundary faces in a two-dimensional domain and four boundaries in a 3-D domain. The remaining two boundary faces are connected implicitly point to point. Therefore, an O-grid has lines of points where the last point wraps around and meets the first point. C-grids have lines of points in one direction which are shaped roughly like the letter “C”, and one of those lines will typically bend back to meet up with itself. An H-grid is any structured grid that is not an O-grid or a C-grid (Kovacevic, 2005).

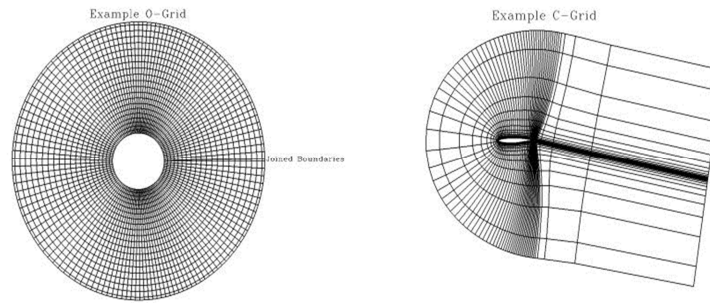


Figure 2.12: Examples of O- and C-grids (Innovative-cfd, 2011) .

2.3.2 Unstructured mesh

One characteristic used to distinguish between structured and unstructured meshes is the existence of an obvious mapping between logical space and physical space for the mesh. Physical space refers to the Cartesian coordinate space used to house the actual mesh (and geometry). Logical space refers to an equivalent domain in which the logical cell-face-node connectivity is represented. If a mapping is available between these two representations, the mesh is structured. If a logical mesh cannot be created for a mesh, then the mesh is unstructured (Hansen et al., 2005).

An unstructured mesh is usually comprised of triangular elements in 2-D and tetrahedral elements in 3-D. There is no presumed order to the mesh so that, given a cell, there is no inherent presumption as to which cells are adjacent to it and no obvious logical relationship to the physical mesh.

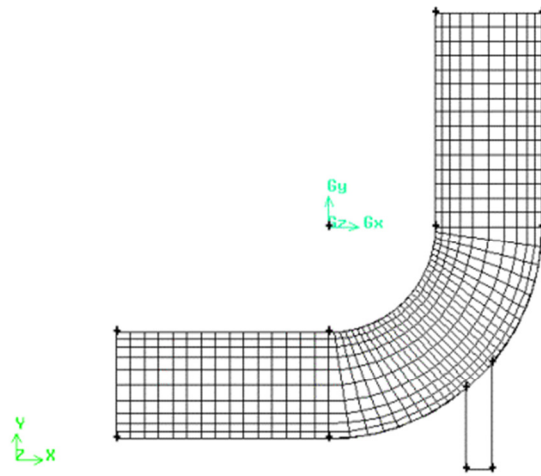


Figure 2.13: Representation of a structured mesh in a cylindrical model.

2.3.3 Structured as opposed to unstructured mesh

A number of research studies have employed a wide variety of mesh styles to assess velocity fields and particle deposition patterns in models of branching biological systems. Depending on the application, structured meshes are known to be superior to unstructured meshes for finite element applications. Although unstructured mesh conform to complex geometries, additional data calculations are needed to connect up arbitrary number of neighbouring nodes, which requires additional computational memory and further complicates the solution algorithms.

De Santis et al. have compared the performance of unstructured and structured meshes in solving the Navier–Stokes equations in a left coronary artery (De Santis et al., 2010). Tetrahedral (unstructured) meshes needed much higher resolution than structured meshes to reach mesh independency, with higher computational costs (computational time and memory). The wall shear stress did plateau with progressive refinement of

structured meshes but not when unstructured meshes were used (De Santis et al., 2011). Such differences were also reported in other studies and are attributed to the high numerical diffusion error associated with unstructured meshes. Unstructured meshes turn out to be ineffective at resolving wall boundary layers since triangles and tetrahedrons do not deform (stretch or bend) during local refinement to make the cells very small. Tetrahedral cells are prone to high aspect ratios which affect the skewness of the cell. In addition, tetrahedral cells can be difficult to align with the principal flow or stress directions which may lead to numerical issues, although in some cases this could be alleviated with higher order discretization schemes and a sufficiently fine mesh. As represented Longest and Vinchurkar, generating structured meshes based on hexahedral elements requires significant time and effort; however, these meshes are often associated with high quality solutions (Longest & Vinchurkar, 2007).

Applications to the human femur have shown that hexahedral meshes should be preferred over unstructured meshes (including quadratic tetrahedrons) in order to improve mesh independency (Viceconti et al., 1998). Ramos and Simões evaluated and compared tetrahedral and hexahedral finite element meshes of simplified and realistic proximal intact femur geometries (Ramos & Simões, 2006). They compared the periosteal von Mises stresses and principal strains at a selected point of the femur in both cases. Overall, they concluded that tetrahedral linear elements allowed results closer to theoretical ones, but hexahedral quadratic elements seem to be more stable and less influenced to the degree of refinement (NDOF) of the mesh.

In the case of the structural analysis of the aorta, for which the material properties are hyperelastic and orthotropic, the available element technology is currently such that

the principal directions of the material are defined by the edges of the hexahedrons (LS-Dyna documentation). Therefore, a hexahedral mesh such that the element edges conform to the local circumferential, longitudinal and radial directions would be most suitable for the analysis.

Based on the above discussion, quad/hexahedral meshes are of interest in the present work for the following reasons:

- A quad/hexahedral meshing of a model results in fewer elements and can approximate complex domains to a close accuracy.
- Quad/hexahedral elements generally deform under lower strain energy states. Thus, hexahedral meshes provide more accurate results when compared to tetrahedral meshes.
- Better numerical behavior is achieved with structured meshes in problems such as stress analysis; they are more stable and less influenced by the degree of refinement of the mesh when compared to tetrahedral meshes.

2.4 Existing FE mesh generation techniques from medical images

Patient specific modeling of blood vessels, including the aorta, appeared in 1990. Its purpose is the investigation of the effects of cardiovascular devices on physiologic function and to predict outcomes of therapies for individual patients (Taylor & Figueroa, 2009). To automate the process of FE model development, different algorithms have been presented using a mapped meshed technique, as presented below.

2.4.1 Early mesh-matching techniques

Couteau et al. proposed a grid-based, mesh matching (M-M) algorithm. The M-M algorithm generates customized 3-D meshes of anatomical structures from an already existing model (Couteau et al., 2000). The M-M algorithm has been applied to create models by matching the tiled surface of the proximal femur with a structured mesh of another femur created manually by an operator. The algorithm uses an adaptive 3-D displacement grid called an octree-spline. The authors applied the M-M algorithm on already existing models, which is not a desirable feature in the present work.

Following the work of other researchers on deformable image registration algorithms, (Brown, 1992; Gaens et al., 1998; Bowden et al., 1998) developed a ‘warping’ method to use segmented image information from a single subject to facilitate the segmentation of anatomic structures of other subjects. They segmented the distal femur and registered neuroanatomical structures by incorporating imaging data directly into a nonlinear FE analysis program. As a continuation of their research, Grosland et al. proposed a method for automated hexahedral meshing (Grosland et al., 2009). They mapped a predefined high quality mesh directly onto a new boney surface with minimum user interaction. To achieve this, they employed a registration technique based on the FE method. This algorithm follows hierarchical models, incorporating multiple levels of mesh refinement. Their algorithm requires an initial template mesh and the desired target surface as inputs. Mapping is a typical procedure for manually controlled mesh generation. It consists in decomposing the object in a sequence of sub-volumes onto which the program maps the mesh (Viceconti et al., 1998).

Chae and Kwon proposed a tetrahedral meshing algorithm from a series of medical images for the femur (Chae & Kwon, 2003). In this scheme, the planar sections of three-dimensional objects and the side surfaces between two planar sections are triangulated first, and then an algorithm is employed to construct tetrahedral elements by using basic operators. A correspondence coefficient is proposed, which is determined by the overlapping area and the distance between two sections. The problem is to generate the best topological adjacency relationships between the points on pairs of contours from adjacent sections by constructing a triangular mesh from their points.

Hu et al. illustrated a point-to-point correspondence between the source and the target shapes using polynomial functions and a distance map. They determined a transformation path which uses a quasi-physical law when blending two topologically identical shapes (Hu et al., 2004). They were able to produce a structured mesh of a boney structure.

Baghdadi et al. used template-base technique to generate FE meshes for different patients (Baghdadi et al., 2005). In their “template-based” meshing method, the finite element volume mesh (tetrahedral) is first manually aligned with the follow-up images. The triangulated surface of the mesh is then automatically deformed to fit the imaged organ boundary. Then, using a Laplacian smoothing algorithm, the deformed meshes are modified to correct triangle (surface nodes) distortion and thus preserve triangle quality.

2.4.2 Ad-hoc techniques

Another image registration technique to generate FE models is presented in (Barber & Hose, 2005). They created an exemplar or reference image, segmented it and then registered to the reference image using an image registration algorithm.

Mansi et al. addressed the finite element modeling of the mitral valve using a machine-learning approach (Mansi et al., 2012). They first estimate the global position of the mitral valve, then detect its landmarks and finally map and deform triangulated surfaces to segment the mitral valve leaflets.

Gonzales et al. introduced methods for generating high-quality bicubic and tricubic Hermite finite element meshes of the atria of the heart with vertices derived from segmented data (Gonzales et al., 2013). They constructed quad meshes manually and then utilised a geometric optimization scheme to deform the left and right atrial surface meshes to the vertices of the smoothed triangular meshes obtained until the root-mean squared (RMS) error was less than a threshold value.

Meyer et al. described a dynamic particle method for constructing isosurface triangulations of sampled, volumetric, three-dimensional scalar fields (Meyer et al., 2007). They used a system of dynamic particles that optimally samples an implicit function such that the particles' relative positions can produce a topologically correct Delaunay triangulation to be used toward mesh generation.

2.4.3 Iso-surface techniques

In iso-surface methods, the location of the surface is determined strictly according to the image intensity. A smooth surface is then produced by placing the points at fractional grid locations, which is defined on the image (Cline et al., 1991). This method is restricted to qualitative applications because the reconstructed surfaces are not accurate due to image inhomogeneities and other artifacts (Foo et al., 1999).

For the purpose of mesh generation, it is generally convenient to obtain an analytical representation of the boundary surface through segmentation. This representation of the surface is where the mesh boundary points will be located. As an example Peiró et al. used boundary representation methods for shape reconstruction and mesh generation of unstructured spatial discretization of patient-specific geometries and for the simulation of flows in these meshes (Peiró et al., 2008). In this approach, the boundary of the object is initially represented as an iso-surface of an implicit function defined in terms of radial basis functions. Applying a marching method, they were able to extract this surface.

2.4.4 Morphing techniques

Generalizing mesh-matching techniques, Sigal et al. illustrated two mesh-morphing algorithms for specimen-specific finite element modeling (Sigal et al., 2008). The method obtains accurate models by adapting a pre-existing geometry to represent a target specimen using morphing techniques. They evaluated the algorithms by measuring the distance between target and morphed geometries and by comparing response to axial loading simulated with FE methods. Their method has some limitations such as the

propagation of any inherent problems existing in the source to the target. In another study conducted by same main researcher (Sigal et al., 2010) they applied morphing techniques to parameterize the geometry of specimen-specific FE models, such that the model shape can be varied in a controlled and systematic way suitable for mesh sensitivity analysis. By extracting the nodes of the baseline model surface and displacing these nodes, they performed the morphing process. They also re-meshed the interior of the morphed geometry. The total displacement applied to the nodes is a linear combination of partial displacements. They demonstrated three techniques for defining the partial displacements: scaling-based, analytic-based deformations and landmark-based deformations.

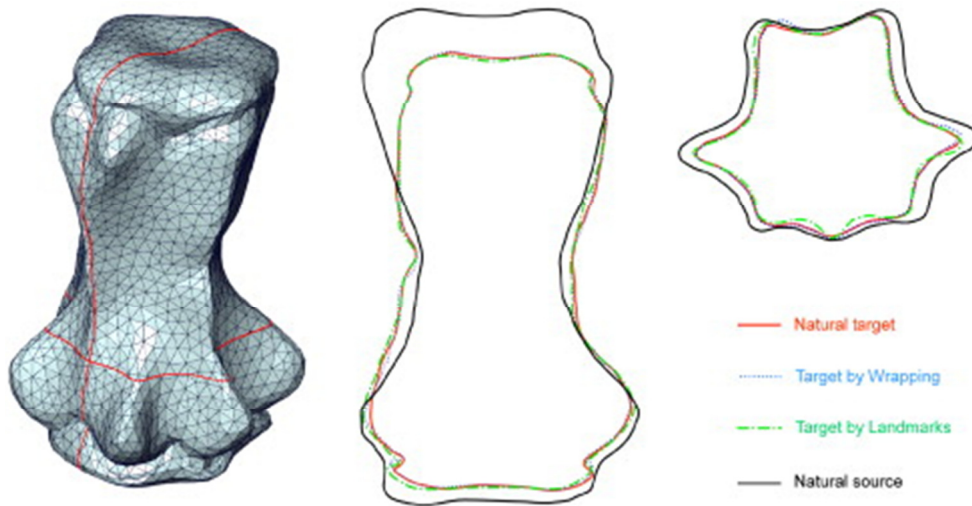


Figure 2.14: Geometry correspondence and morphing: a) Surface of the target model, b) Comparison of coronal (center) and sagittal (right) cross sections of the target model (red line), (Sigal et al., 2008).

2.4.5 Deformable-model based techniques

The deformable model techniques tightly link the description of the object elements (i.e. triangles) to the way it can be deformed. The most simple reconstruction methods deform a triangular mesh by directly estimating the displacements of the triangle vertices. A number of researchers have incorporated the knowledge of a target object shape into a deformable object by using a deformable shape template. The use of anatomical knowledge to guide shape recovery is important for robust and automatic interpretation of medical images (Battle et al., 1997). Bardinet et al. fitted a deformable super quadric to segmented three- dimensional cardiac images and then refined the superquadric fit using a volumetric deformation technique known as free-form FFD (Bardinet et al., 1995).

Magee et al. combined 3-D deformable models and level set methods to mesh an abdominal aortic aneurysm (AAA) (Magee et al., 2001). Their initial model is a triangulated mesh which is deformed at each iteration. The deformation is controlled by the curvature of the model surface.

Using deformable surface models Putter et al. generated tetrahedral meshes for vascular structures. The 3-D deformable models iteratively deformed to fit the shape to image features. During segmentation the new position for a node is determined at every iteration from the current position via a time-discretised second order Newtonian-evolution equation (Putter et al., 2006).

Shim et al. defined an effective processing technique to reconstruct a 3-D shape of an AAA from (CT) images. They deformed the template geometry by extended free-form deformation (EFFD), to generate a patient-specific AAA geometry (Shim et al. 2009).

This method builds a shape for structural finite element analyses and computational fluid dynamics for medical diagnosis.

Auer and Gasser used a hierarchical technique to generate FE meshes for a structural analysis of AAAs from standard clinical CT-A data. They used 2-D and 3-D deformable models to segment the images, and discretized the aneurysm wall into an hexahedral mesh (Auer & Gasser, 2010). Their approach requires minimal user input. Following their work De Santis et al. built structured hexahedral meshes for vascular modeling using topologically equivalent block-structures that are placed inside and outside of the lumen domain (De Santis et al., 2011). They fitted a grid of multi-blocks inside the lumen surface and then series of layers were projected from the surface of the grid.

Ayyalasomayajula et al. used 3-D active contour models for reconstruction and segmentation of AAA (Ayyalasomayajula et al., 2010). They used an iterative algorithm to deform the initial spherical surface to the final segmented geometry.

The other researcher, Kovacs, generated a rough aortic mesh by first approximating the aortic arch and descending aorta and then based on the detected and interpolated circles along the aortic centerline, the surface mesh of the entire aorta is generated by sampling and connecting the neighbouring circles with triangles (Kovacs, 2010).

When it comes to meshing bifurcations, many existing techniques require the existence of a 3-D surface mesh of a bifurcating area of interest. It is typically obtained from a deformable model approach. In their image-based computational modeling of

vascular segments, Antiga and Steinman decompose and map bifurcating vessels using the already built-in surface made up of linear triangular elements (Antiga & Steinman, 2004).

2.4.6 Existing software packages

Parallel to the existing methods in the literature, there are open source packages, including the Vascular Modeling Toolkit (VTK), for 3-D reconstruction and geometric analysis of image-based modeling of blood vessels (GitHub, 2013). Another package is pyFormex that generates, transforms and manipulates large geometrical models of 3-D structures (Verhegghe, 2010). Both software packages can be used for segmentation, which is a prerequisite to generating a FE mesh in the current approach.

Natarajan et al. used the IA-FEMesh software to map building block structures and mesh the surface of a femur (Natarajan et al., 2010). They then morphed the mesh seeds to the bony surface to create a mesh that is composed solely of hexahedral elements. The IA-FEMesh open source software was also used by Shivanna et al. (Shivanna et al., 2010).

2.5 Engineering analysis of the aorta

At this juncture in the presentation, it is timely to more explicitly describe what types of FE analyses can be carried out on the aorta.

2.5.1 Solid mechanics analyses

In solid mechanics analyses, the researchers are mostly interested in evaluating the stress distribution in the aortic wall. They typically compare the local stress values to the local strength values for the wall to assess the risk of rupture of the aorta, as occurs in aneurysms. The analyses are performed based on a series of CT or MRI images captured at a particular time during the heart cycle. Therefore, the mesh needs only to be obtained once, and may be considered static.

2.5.2 Fluid mechanics analyses

In this type of analyses, fluid velocity vectors and pressures are calculated throughout the domain of interest, constrained by boundary conditions. The latter typically involve no-slip boundary conditions at the vessel wall and in- and outflow boundary conditions at the inlet and outlet of the vessel. Shear stress in the blood near the aortic wall is the focus, as an alleged culprit for atherosclerosis development. These studies require a refined mesh close to the boundaries. These boundaries are moving as the blood flow is pulsatile and the aortic walls move accordingly. Therefore, information about the boundaries at different points in time during the cardiac cycle is needed to be able to do flow analysis, except in cases where flow is steady, such as with left-ventricular assist devices (rotary pumps designed to help the heart perform its function). Fluid meshes can be derived from different medical images such as CT, MR, or ultrasound images (Antiga et al., 2008).

2.5.3 Fluid-structure interaction analyses

Fluid-structure interaction (FSI) analyses look at the interaction between fluid flow and the deformable structure that contains it. Flow creates pressure on the solid boundaries; the pressure deforms the boundaries which, in turn, modify the flow. Therefore, both the

aortic wall and the fluid need to be meshed. Fluid–structure interactions can be stable or oscillatory.

Wolters et al. presented a patient-specific computational model for FSI in AAA (Wolters et al., 2005). To build their model, they used a standardized bifurcated mesh that is transformed based on the centerlines and the boundary surface of the segmented AAA.

3 Three-dimensional gradient vector flow

3.1 Formulation

Since we aim to use features of the gradient vector flow in combination with 3-D images, it is imperative to clearly formulate the GVF in 3-D. Ayyalasomayajula et al. illustrated the overall algorithm to construct the GVF in 3-D, but failed to explain the origin of their equations (Ayyalasomayajula et al., 2010). We derived the following as an extension of the 2-D formulations found in the literature.

In 3-D, the gradient vector flow field $\mathbf{G}$ is introduced as an approximation of the gradient of a known 3-D function f , and is defined as the minimum of the energy functional:

$$E(\mathbf{G}, f) = \iiint \mu \|\nabla \mathbf{G}\|^2 + \|\nabla f\|^2 \|\mathbf{G} - \nabla f\|^2 dx dy dz \quad (3.1)$$

where $\nabla f = \begin{Bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{Bmatrix}$ is the gradient of function f .

Parameter μ is known as a regularization parameter governing the trade-off between the first term and second term in the energy functional in (3.1). When the value of the gradient ∇f is small, the energy functional is dominated by the sum of the partial derivatives of the gradient vector field, and its minimum yields a smooth field. Conversely, when ∇f is large, the second term dominates the integrand in (3.1). In this

case, setting $\mathbf{G} = \nabla f$ minimizes the energy. Overall, this formulation produces a field which is equal to the gradient ∇f where the gradient is large, and varies smoothly in regions of the domain where the gradient is small.

Based on the theory of variations, stationary values of the energy functional $E(\mathbf{G}, f) = \int_{\Omega} \mathcal{L} \left(x, y, z, \mathbf{G}, \frac{\partial \mathbf{G}}{\partial x}, \frac{\partial \mathbf{G}}{\partial y}, \frac{\partial \mathbf{G}}{\partial z} \right) d\Omega$, where $\mathcal{L}$ is a real-valued function, and $\mathbf{G}$ is a 3-D vector function of variables x , y and z , are obtained when the following Euler-Lagrange equation is satisfied (Courant & Hilbert, 1953):

$$\frac{\partial \mathcal{L}}{\partial \mathbf{G}} - \frac{\partial}{\partial x} \left(\frac{\partial \mathcal{L}}{\partial \left(\frac{\partial \mathbf{G}}{\partial x} \right)} \right) - \frac{\partial}{\partial y} \left(\frac{\partial \mathcal{L}}{\partial \left(\frac{\partial \mathbf{G}}{\partial y} \right)} \right) - \frac{\partial}{\partial z} \left(\frac{\partial \mathcal{L}}{\partial \left(\frac{\partial \mathbf{G}}{\partial z} \right)} \right) = \mathbf{0}.$$

From (3.1),

$$\mathcal{L} = \mu \|\nabla \mathbf{G}\|^2 + \|\nabla f\|^2 \|\mathbf{G} - \nabla f\|^2 \quad (3.2)$$

where $\mathbf{G} = \begin{pmatrix} u(x, y, z) \\ v(x, y, z) \\ w(x, y, z) \end{pmatrix}$ is unknown.

Noting that:

$$\|\nabla \mathbf{G}\|^2 = \left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial w}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 + \left(\frac{\partial w}{\partial y} \right)^2 + \left(\frac{\partial u}{\partial z} \right)^2 + \left(\frac{\partial v}{\partial z} \right)^2 + \left(\frac{\partial w}{\partial z} \right)^2,$$

it follows that

$$\mathcal{L} = \mu \left(\left\| \frac{\partial \mathbf{G}}{\partial x} \right\|^2 + \left\| \frac{\partial \mathbf{G}}{\partial y} \right\|^2 + \left\| \frac{\partial \mathbf{G}}{\partial z} \right\|^2 \right) + \|\nabla f\|^2 \|\mathbf{G} - \nabla f\|^2. \quad (3.3)$$

Then,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{G}} = 2\|\nabla f\|^2 (\mathbf{G} - \nabla f); \quad (3.3)$$

$$\frac{\partial \mathcal{L}}{\partial \left(\frac{\partial \mathbf{G}}{\partial x}\right)} = 2\mu \frac{\partial \mathbf{G}}{\partial x} = 2\mu \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial x} \\ \frac{\partial w}{\partial x} \end{Bmatrix}; \quad \frac{\partial \mathcal{L}}{\partial \left(\frac{\partial \mathbf{G}}{\partial y}\right)} = 2\mu \frac{\partial \mathbf{G}}{\partial y} = 2\mu \begin{Bmatrix} \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial y} \\ \frac{\partial w}{\partial y} \end{Bmatrix}; \quad \frac{\partial \mathcal{L}}{\partial \left(\frac{\partial \mathbf{G}}{\partial z}\right)} = 2\mu \frac{\partial \mathbf{G}}{\partial z} = 2\mu \begin{Bmatrix} \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial z} \end{Bmatrix}.$$

Therefore, the Euler-Lagrange equations associated with (3.2) are:

$$\begin{cases} \|\nabla f\|^2 \left(u - \frac{\partial f}{\partial x}\right) - \mu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) = 0 \\ \|\nabla f\|^2 \left(v - \frac{\partial f}{\partial y}\right) - \mu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right) = 0 \\ \|\nabla f\|^2 \left(w - \frac{\partial f}{\partial z}\right) - \mu \left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) = 0 \end{cases} \quad (3.4)$$

Or, in other words,

$$\|\nabla f\|^2 (\mathbf{G} - \nabla f) - \mu \nabla^2 \mathbf{G} = \mathbf{0}. \quad (3.5)$$

The full derivation of the numerical method used for solving these equations is shown in Appendix A.

3.2 Implementation of 3-D GVF field using synthetic images

We implemented the numerical method to determine the GVF field in 3-D using images generated synthetically (referred to in the following as synthetic images). The advantage of synthetic images is twofold: their content can be precisely defined, and noise can be introduced or removed at will. To evaluate the properties of the GVF field in 3-D, noiseless images were generated in the form of a hollow cylinder following a curved centerline. The volume generation procedure is shown in Figure 3.1. Consistent with a normal aorta, the radius of the cylinder was 12 mm, the curvature of the centerline in the aortic arch was 40 mm and the wall was approximately two voxels thick. The voxel size

was 1.0 by 1.0 by 1.0 mm. All the calculations and simulations in this thesis were performed in MATLAB R2011b (The MathWorks, Natick, MA, USA) on a Windows Server 2008 workstation with two Intel Xeon E5640, 2.60 GHz, 4-core processors and 32 GB of RAM. Only 5 GB of RAM was used, at the most, in all applications.

Input: aortic minor radius (r), aortic major radius (R)
Output: 3-D image intensity matrix for synthetic aorta (`images`)

```
function images = ImGenerator(r,R)
    % Size of 3-D image domain
    nvoxX = round(20/3*r);
    nvoxY = round(70/6*r);
    nvoxZ = round(25/3*r);

    % Initialization of 3-D images: 0 = black color
    images = zeros(nvoxX,nvoxY,nvoxZ,'int16');
    for i = 1:1:nvoxX
        for j = 1:1:nvoxY
            % Straight segment of aorta (descending thoracic)
            for k = 1:1:27
                res = (j-0.5*nvoxY-R)^2+(i-0.5*nvoxX)^2-r^2;
                if abs(res) <= 16
                    images(i,j,k) = 1; % 1 = white color
                end
            end
            % Curved segment of aorta (arch and ascending aorta)
            for k = 28:1:nvoxZ
                res = ((j-0.5*nvoxY)^2+
                    +(i-0.5*nvoxX)^2+
                    +(k-28)^2+R^2-r^2)^2
                    - 4*R^2*((j-0.5*nvoxY)^2+(k-28)^2);
                if abs(res) <= 90000
                    images(i,j,k) = 1; % 1 = white color
                end
            end
        end
    end
end
```

Figure 3.1: MATLAB code to create a synthetic volume image of the aorta.

The algorithm presented in Figure 3.1 is capable of generating a synthetic aortic shape volume and its centerline coordinates. Once the volume is generated, we can generate the 3-D GVF field associated with it.

The algorithm for 3-D GVF calculation is based on the original 2-D implementation by (Xu & Prince, 1998), the formulations described in the previous section and detailed derivations in Appendix A. The inputs are the 3-D image, the regularization parameter and the desired iteration number. Depending on the amount of noise, the regularization parameter can be adjusted. The more the noise, the higher the values of the regularization parameter should be. For the case of synthetic images, the regularization parameter was set to 0.2. The number of iterations can be adjusted to control the capture range of the GVF field. More iterations increase the capture range.

After the volume image is normalized, the image matrix is expanded using mirror boundary conditions. This function applies mirror boundary conditions on the 3-D input image reflecting on left and right boundary corners. The values of all voxels at the boundaries are set to the values of the voxels 2 steps inward. The GVF values are then initialized as those of the image gradient. They are subsequently updated in a for-loop for the number of iterations. The next stage consists in shrinking the expanded image matrix to carry it back to the original format. Therefore the values of all voxels at the boundary are set to the values of the voxels 2 steps outward. Finally, the GVF values are normalized to produce values in the range of zero to one.

The GVF field generation algorithm is presented in Figure 3.2.

Input: 3-D image intensity matrix (*images*), regularization parameter (*mu*), number of iterations for GVF computation (*iter*)

Output: 3-D GVF field components (*u, v, w*)

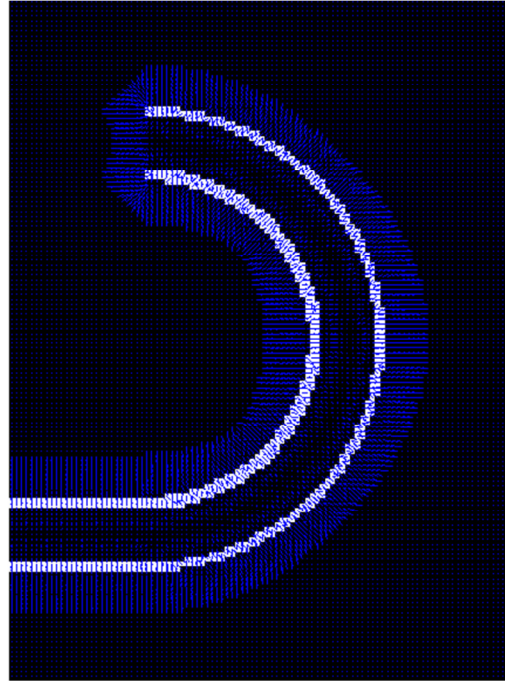
```
function GVF(images,mu,iter)
    f = images;
    fmin = min(f(:));
    fmax = max(f(:));
    f = (f-fmin)/(fmax-fmin); % Normalize f
    f = BoundMirrorExpand(f); % Boundary conditions
    [fx,fy,fz] = gradient(f); % Gradient of the images
    u = fx; v = fy; w = fz; % Initialize GVF as the gradient
    SqrMagf = fx.*fx + fy.*fy + fz.*fz;
    % Iteratively solve for the components u,v,w of GVF
    for i = 1:iter
        u = u + mu*6*del2(u) - SqrMagf.*(u-fx);
        v = v + mu*6*del2(v) - SqrMagf.*(v-fy);
        w = w + mu*6*del2(w) - SqrMagf.*(w-fz);
    end
    u = BoundMirrorShrink(u);
    v = BoundMirrorShrink(v);
    w = BoundMirrorShrink(w);
    % Normalize GVF
    mag = sqrt(u.*u + v.*v + w.*w);
    u = u./(mag + 1e-10);
    v = v./(mag + 1e-10);
    w = w./(mag + 1e-10);
    function B = BoundMirrorExpand(A)
        [m,n,p] = size(A);
        yi = 2:m+1;
        xi = 2:n+1;
        zi = 2:p+1;
        B = zeros(m+2,n+2,p+2);
        B(yi,xi,zi) = A;
        % Mirror corners
        B([1 m+2],[1 n+2],[1 p+2]) = B([3 m],[3 n],[3 p]);
        % Mirror left and right boundaries
        B([1 m+2],xi,zi) = B([3 m],xi,zi);
        B(yi,[1 n+2],zi) = B(yi,[3 n],zi);
        % Mirror top and bottom boundaries
        B(yi,xi,[1 p+2]) = B(yi,xi,[3 p]);
    end
    function B = BoundMirrorShrink(A)
        [m,n,p] = size(A);
        yi = 2:m-1;
        xi = 2:n-1;
        zi = 2:p-1;
        B = A(yi,xi,zi);
    end
end
```

Figure 3.2: MATLAB code for 3-D GVF field computation.

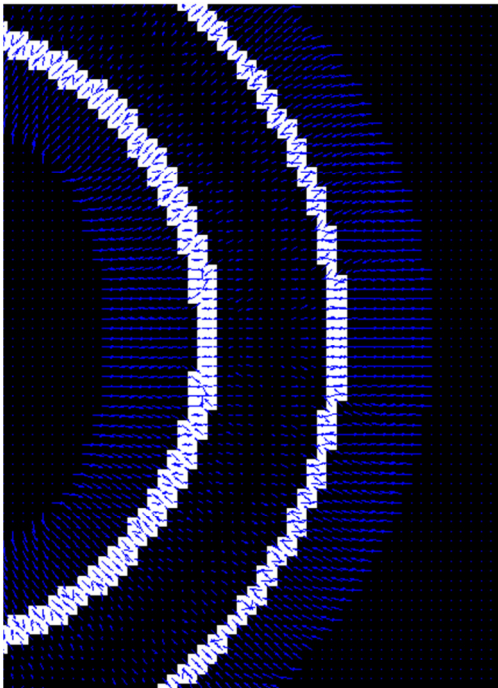
Figure 3.3 presents slices of the original synthetic aorta and the corresponding 3-D GVF field.



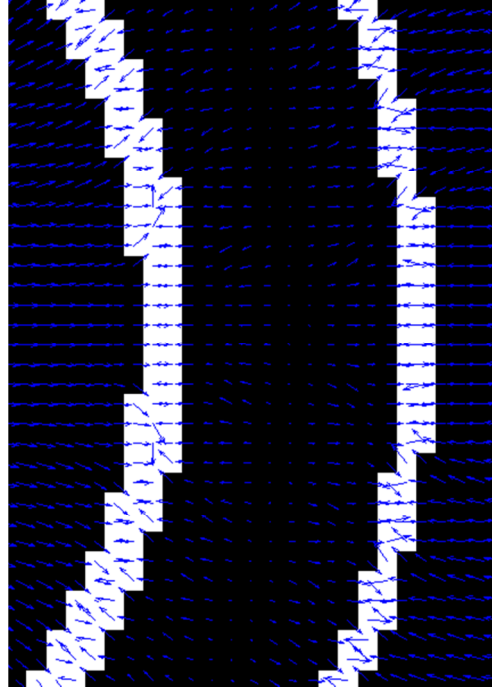
(a)



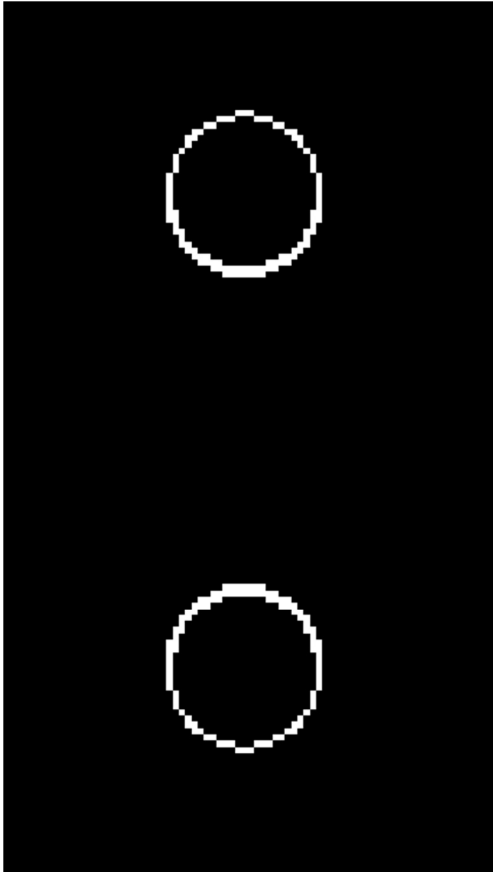
(b)



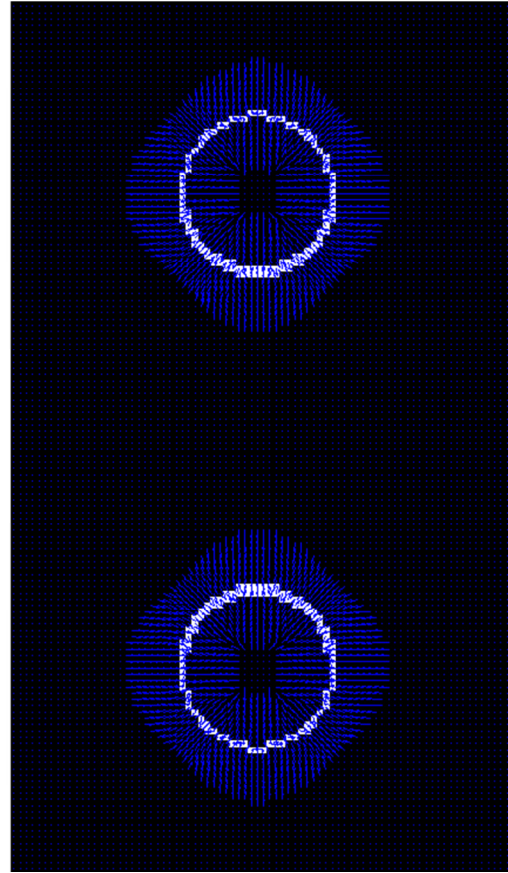
(c)



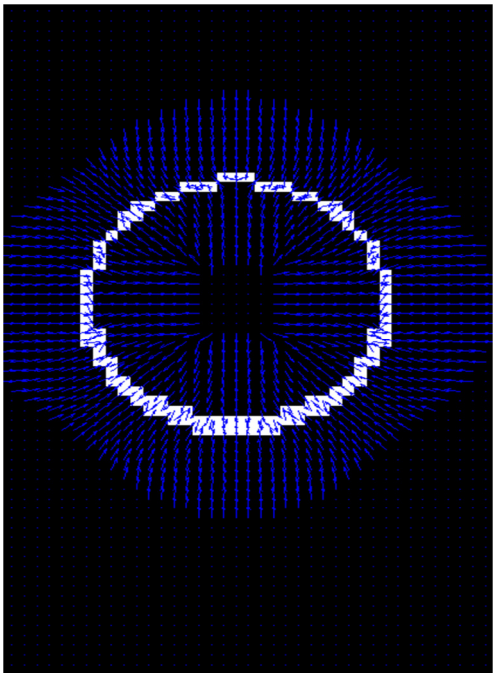
(d)



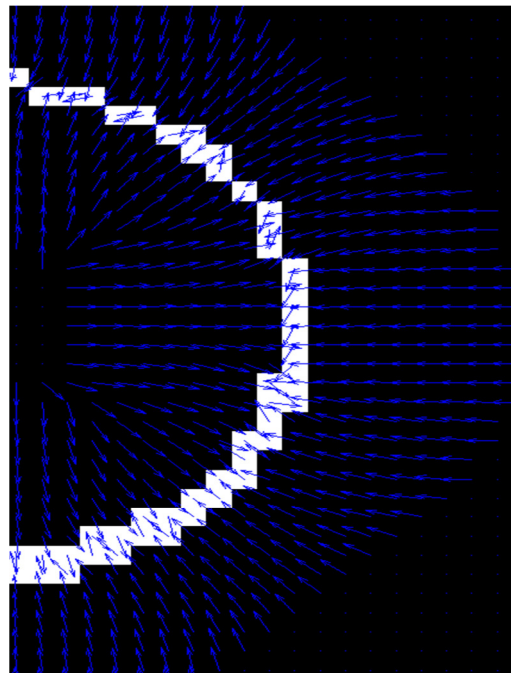
(e)



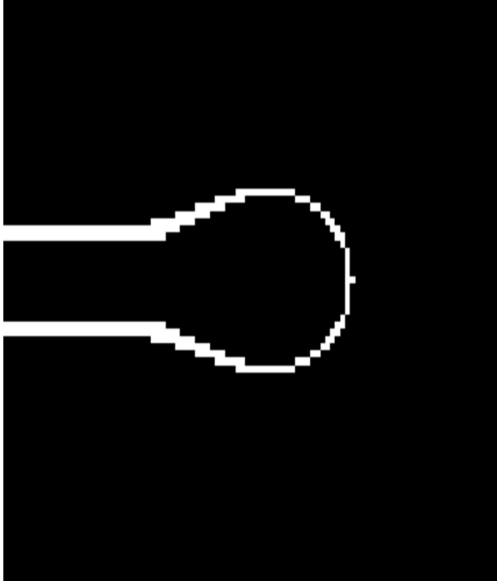
(f)



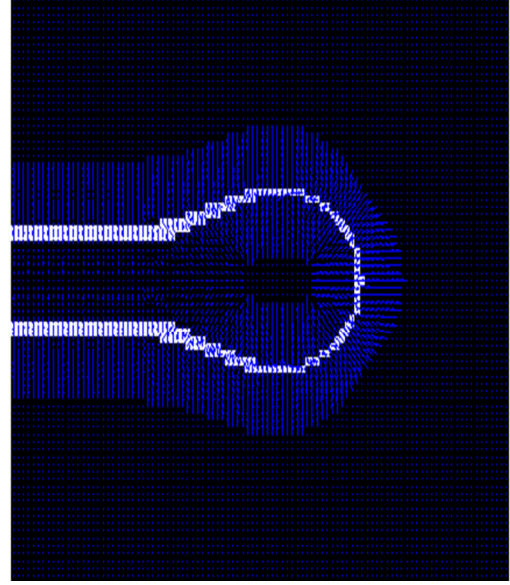
(g)



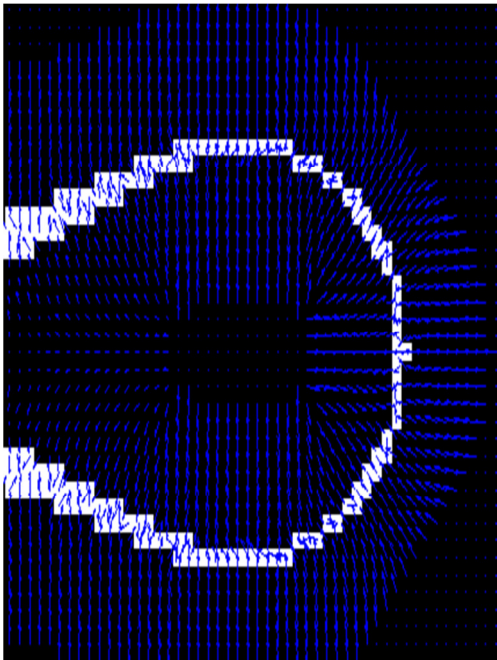
(h)



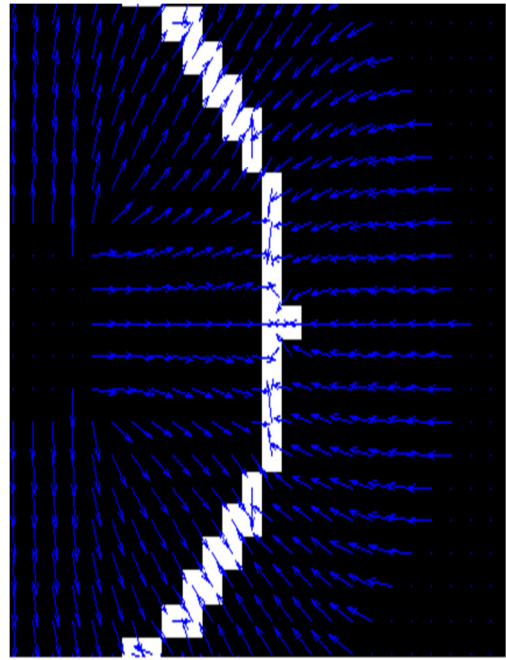
(i)



(j)



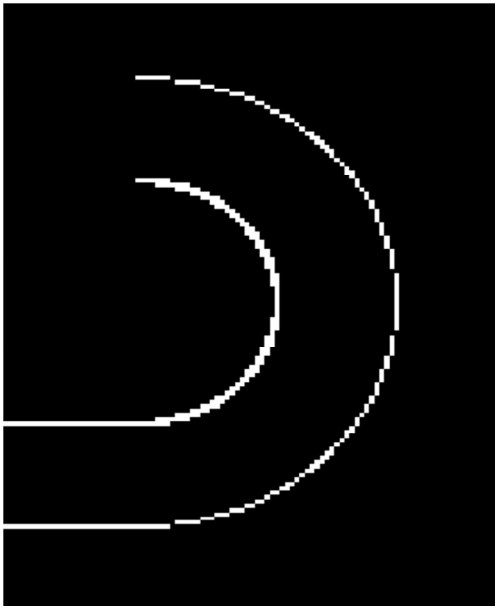
(k)



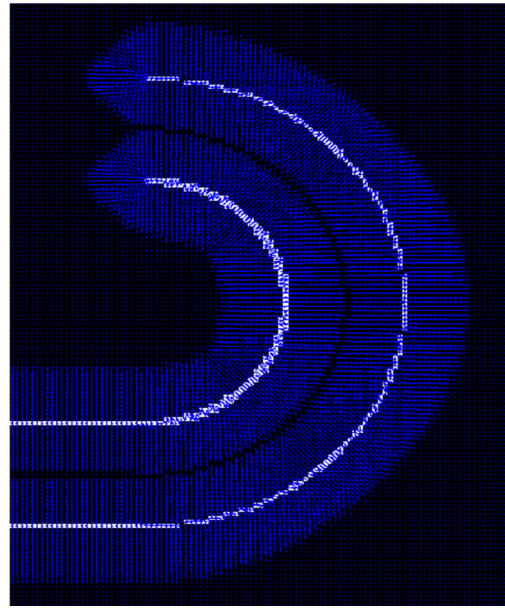
(l)

Figure 3.3: Implementation of 3-D GVF on synthetic aortic images; (a), (e), (i); Different cross sections of the synthetic images; (b), (f), (j) GVF field overlaid on top of the original images; (c), (d), (g), (h), (k), (l) Magnified GVF field overlaid on top of the original images-magnified, after 8 iterations, with $\mu=0.2$.

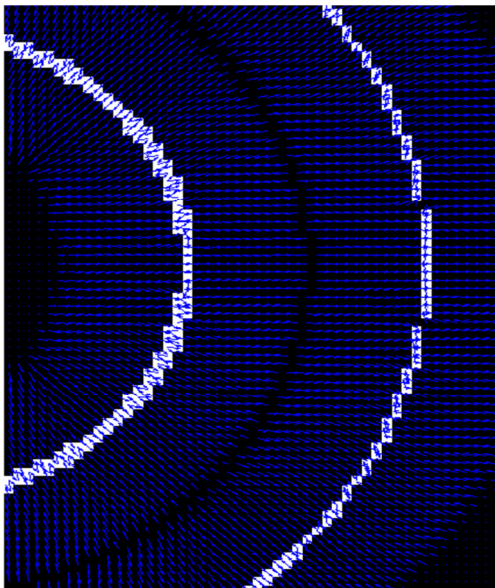
As shown in Figure 3.3, the vectors of the 3-D GVF field are pointed toward the wall of the cylinder, which confirms the usefulness of the GVF as a segmentation tool. In addition, the capture range of the GVF (i.e. how far an edge produced vectors) was easily increased by increasing the number of iterations in the computation. For instance, fifteen iterations were enough to have the GVF field start from the center of the aorta (Figure 3.4).



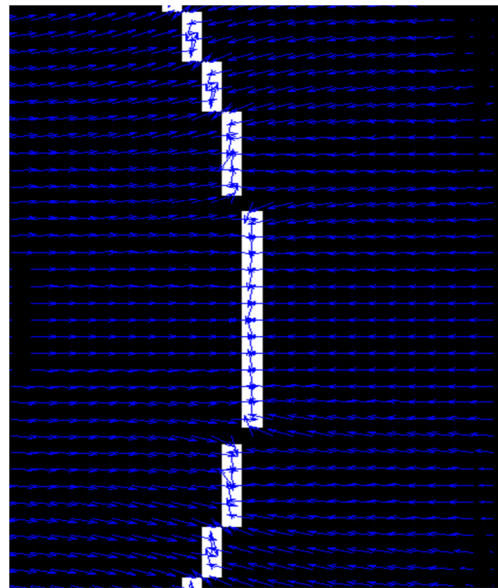
(a)



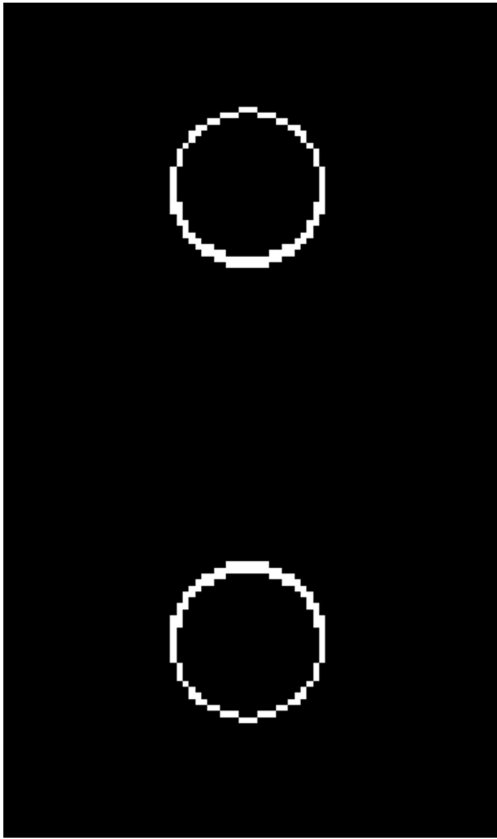
(b)



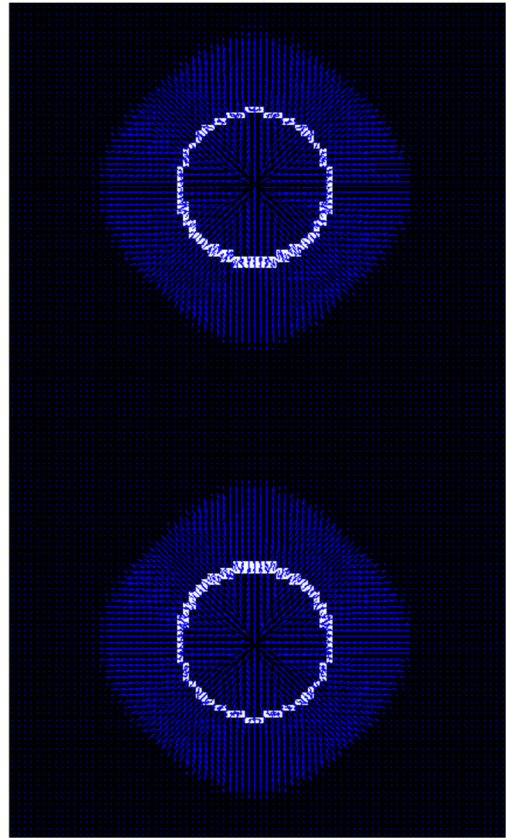
(c)



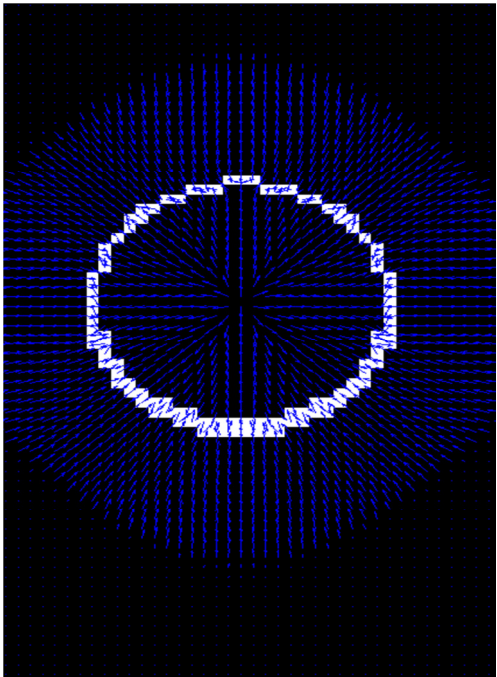
(d)



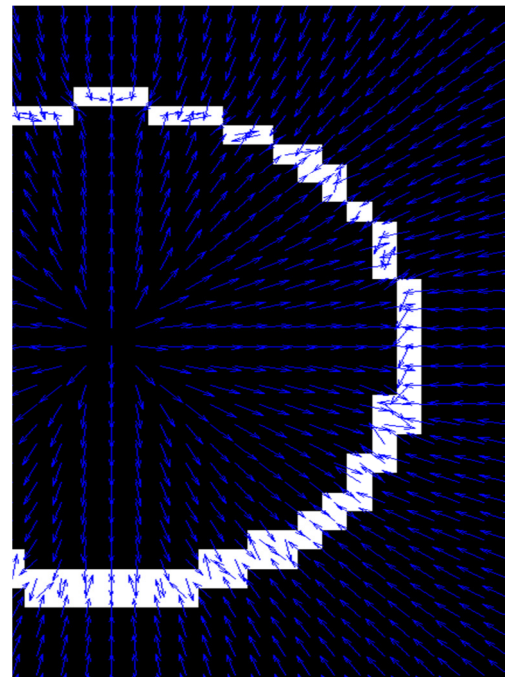
(e)



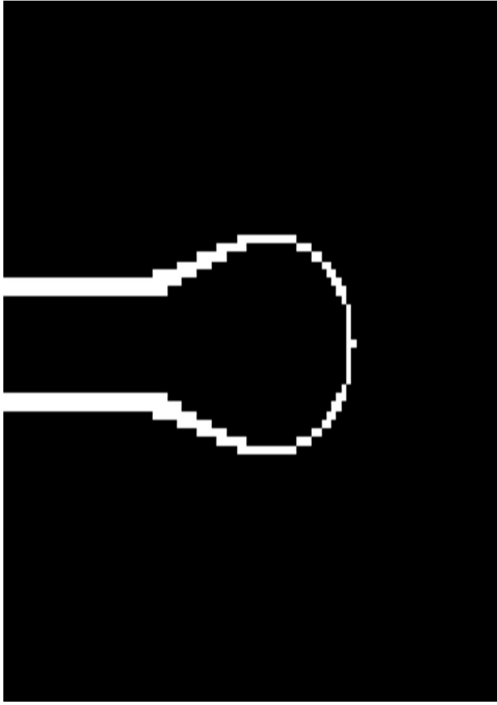
(f)



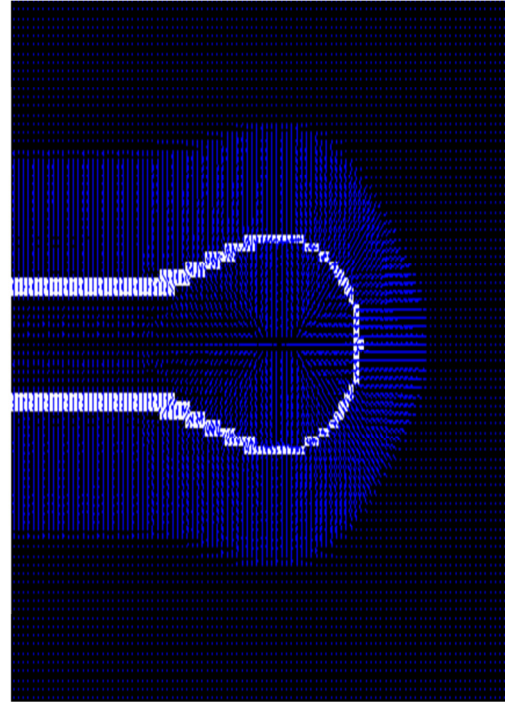
(g)



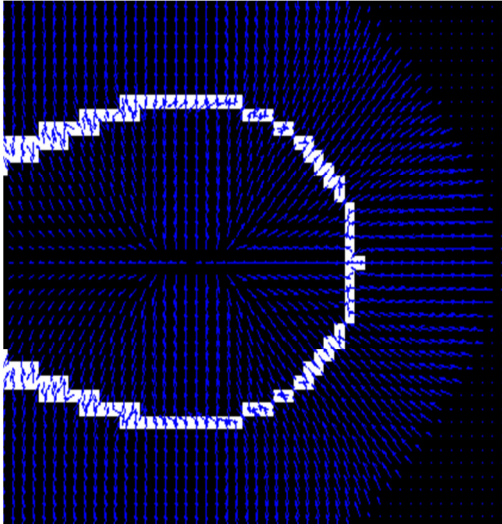
(h)



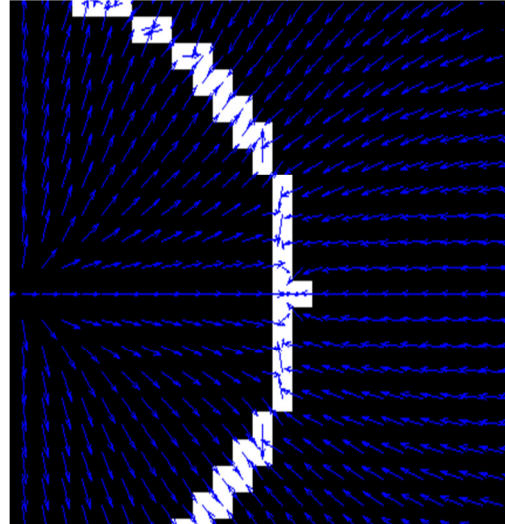
(i)



(j)



(k)



(l)

Figure 3.4: Implementation of 3-D GVF on synthetic images; (a), (e), (i); Different cross sections of the synthetic images; (b), (f), (j) GVF field overlaid on top of the original images; (c), (d), (g), (h), (k), (l) Magnified GVF field overlaid on top of the original images-magnified, after 15 iterations with $\mu=0.2$.

The capture range of the 3-D GVF was easily increased by increasing the number of iterations at minimal processing cost (time). This property will be beneficial when dealing with medical images where a variety of objects/organs exist in the same neighborhood and consequently the vector field must be large enough to contain all the boundaries of interest.

3.3 Implementation of 3-D GVF field using medical images

To evaluate the performance of the 3-D GVF field in medical images, we used de-identified data obtained from a Toshiba helical CT scanner that were made available to us by Dr. C. Beller from the University of Heidelberg, Germany. The voxel size of the dataset was 0.3 by 0.3 by 0.3 mm. Since the GVF field is sensitive to image noise, it quickly became apparent that some filtering of the data was necessary for the GVF to perform well. Filtering was introduced in the form of a classic Canny edge detector, as presented below.

3.3.1 Canny-edge detection

The main purpose of using edge detection is to reduce the amount of data in images and increase contrast while preserving the structural properties to be used for further image processing. This makes it possible to stay with reasonable processing times (e.g. on the order of minutes) while minimizing memory issues. In a typical 2-D Canny edge detector, a Gaussian convolution is first used to smoothen the image; then, a simple 2-D first derivative operator is applied to the smoothed image to highlight regions of the image with high first spatial derivative (Canny, 1986; Fisher et al., 2003).

To adapt these principles to 3-D images, the process was repeated in 3 orthogonal planes, and the sum of the 3 results constituted the filtered 3-D images. The 3 orthogonal planes correspond to the anatomic planes shown in Figure 3.5 as imaged by a medical scanner.

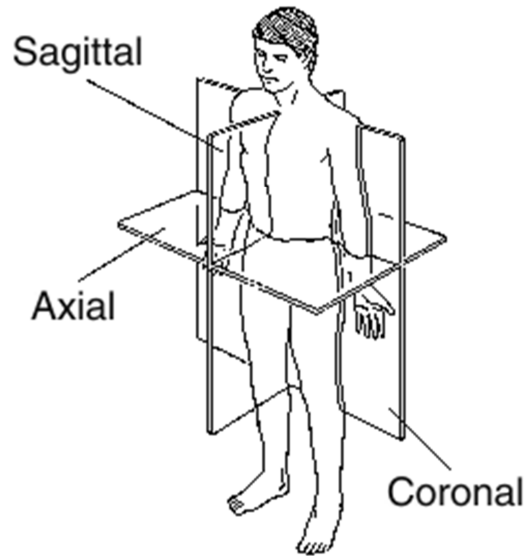
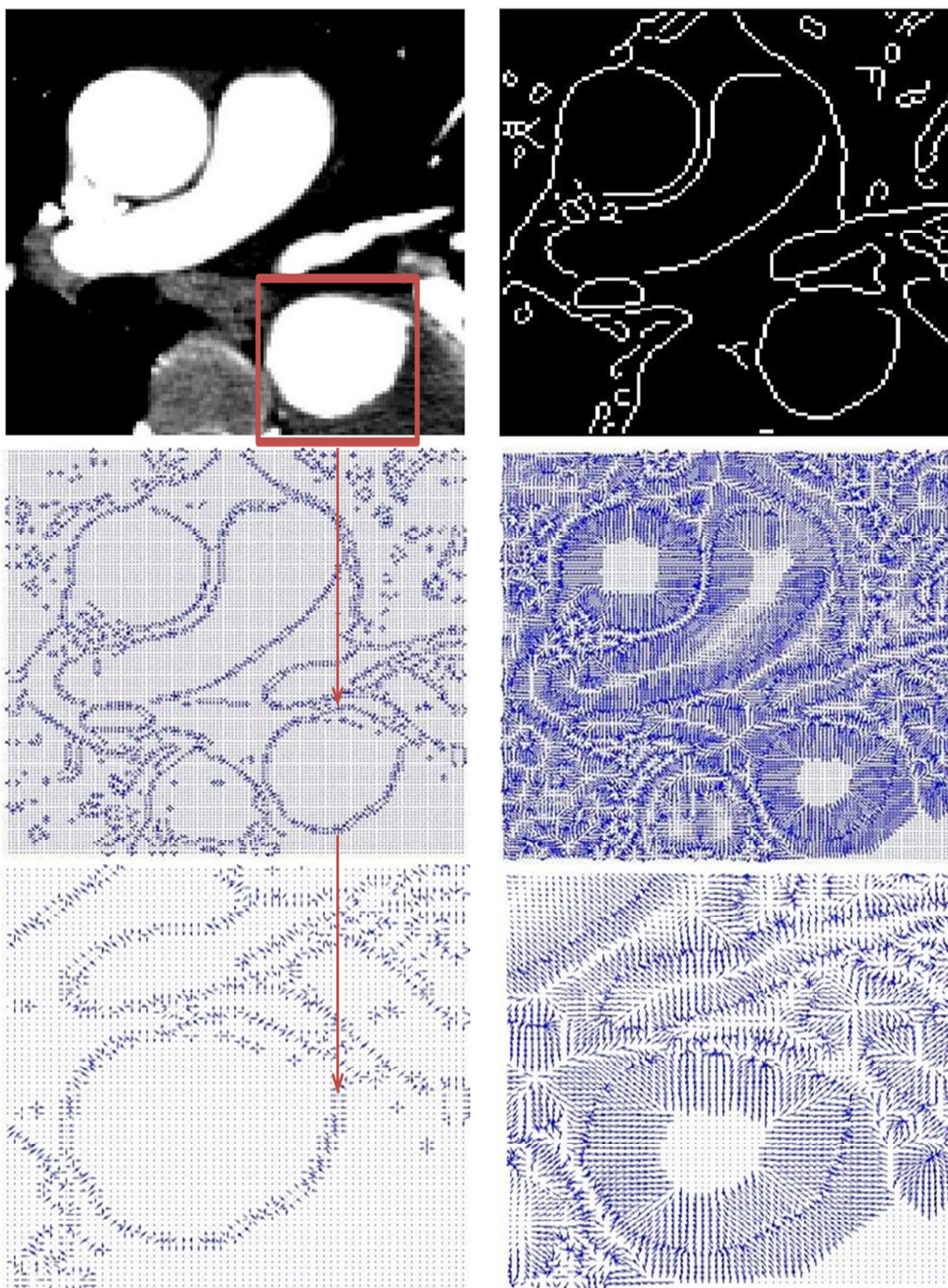


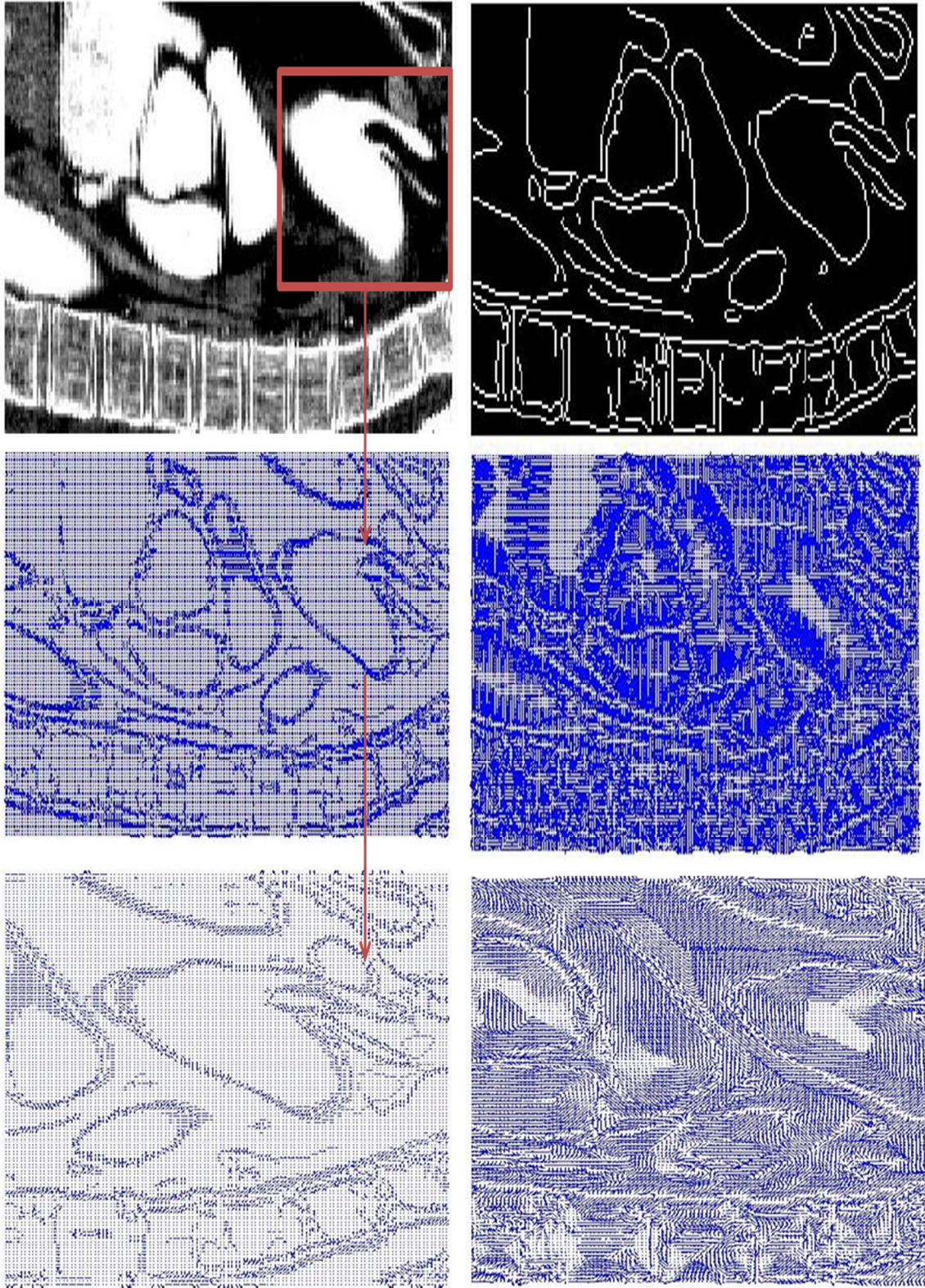
Figure 3.5: Anatomical planes.

3.3.2 GVF field for CT images

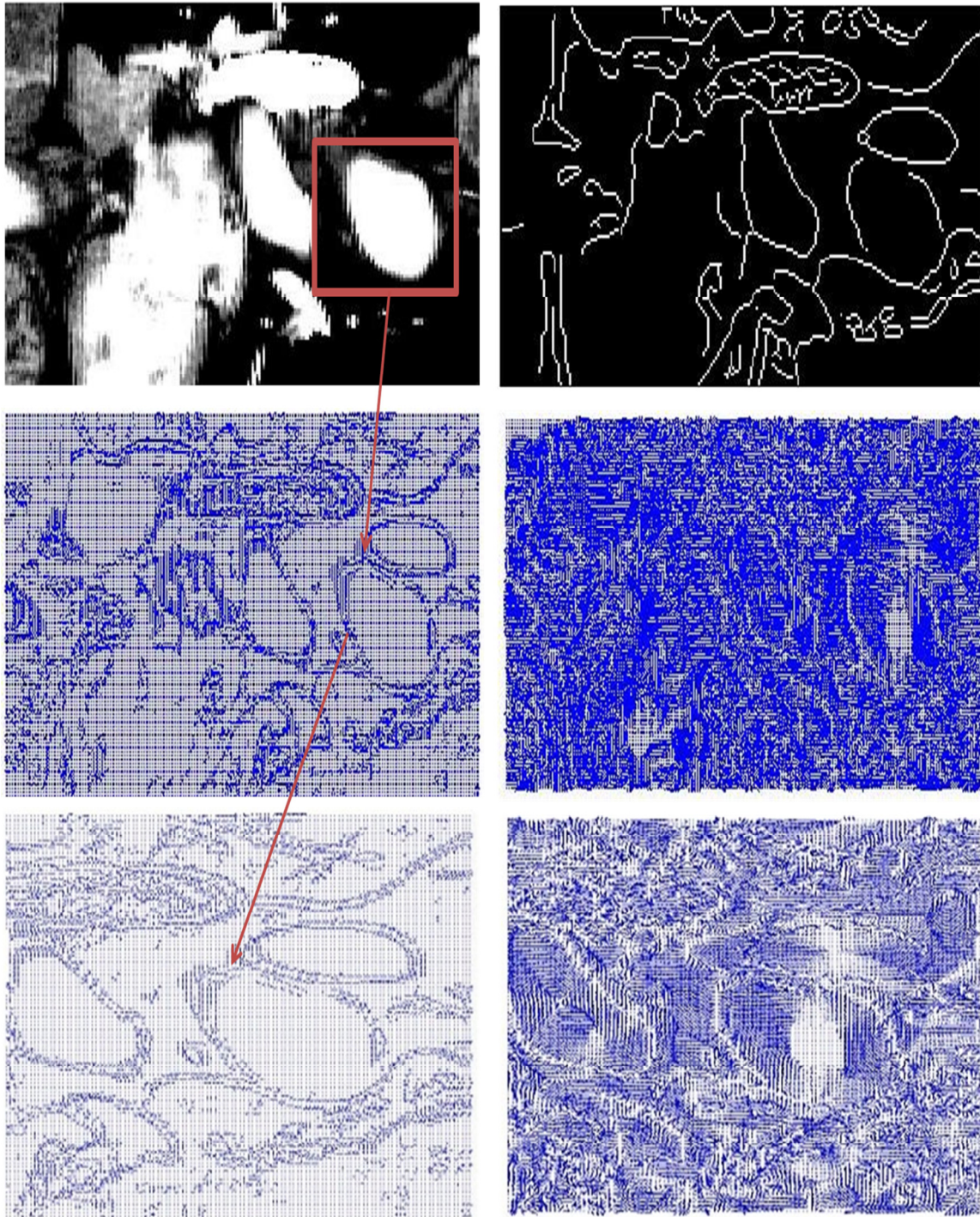
After pre-processing the images using the Canny edge detector, the 3-D GVF field was determined using the same algorithm as discussed earlier. The parameters of the GVF were as follows: regularization parameter = 0.2; number of iterations = 125. The results are illustrated in Figure 3.6 for a dataset introduced later in Section 5.9.



(a)



(b)



(c)

Figure 3.6: Left: top: Original image; middle and bottom: Gradient image; Right: top: Edge map; middle and bottom: GVF field. (a) Axial, (b) Coronal, (c) Sagittal.

The GVF field in regions of interest is shown in Figure **3.6** and magnified to emphasize the direction of the vectors. As can be seen, the vectors point to the borders of the organs' contours in all cases, again confirming the implicit segmentation powers of the GVF, even in 3-D.

To try and harness these features to effectively and simultaneously segment and mesh a 3-D aorta proved a challenging and elusive proposition. To illustrate the evolution of our thinking, the next chapter details attempts at using the 3-D GVF field to deform an arbitrary cylindrical finite element surface mesh until it matches the shape of a 3-D aorta. Although these attempts failed, they have value as a learning experience and indeed contributed to guiding us towards a successful method.

4 Direct finite element implementation and a variant

In this chapter, we explore the possibility of using the 3-D GVF field as a true force field that can deform a structure. The structure investigated was an arbitrary cylindrical structured mesh to be deformed into the synthetic aortic shape. The details of this implementation are given first. After evaluation, a variant of the method is proposed, wherein the 3-D GVF field is interpreted as an imposed displacement field.

4.1 Creation of an arbitrary structured mesh

To test the direct FE implementation on the synthetic aortic shape, we simply used the known centerline of the aorta to construct an arbitrary cylindrical structured mesh.

In this algorithm shown in Figure 4.1, the surface nodes are generated based on the centerline coordinates known from the creation of the synthetic aorta. Although this did not correspond to a real situation where the centerline of a patient's aorta would not be known a priori, this was as simple testing setup.

Inputs: 3-D image intensity matrix (*images*), number of cross-sections along cylinder (*n*), aortic major radius (*R*), number of nodes around aortic circumference of cylinder (*ncirc*)

Output: Initial coordinates of nodes on cylindrical structured mesh (*xn*, *yn*, *zn*)

```
% Arbitrary centerline of synthetic aorta (as in Figure 3.1)
[nvoxX,nvoxY,nvoxZ] = size(images);
WPX = 0.5*nvoxX*ones(1,n);
WPY = 0.5*nvoxY*ones(1,n);
WPZ = zeros(1,n);
for i = 1:27
    WPX(1,i) = 0.5*nvoxX+R;
    WPZ(1,i) = i;
end
for i = 28:1:nvoxZ
    WPX(1,i) = 0.5*nvoxX + R*cos((i-28)*pi/(nvoxZ-28));
    WPZ(1,i) = 28 + R*sin((i-28)*pi/(nvoxZ-28));
end
XYZ = [WPX;WPY;WPZ]; % Centerline points (coordinates in voxel space)
normal1 = ones(1,n);
normal2 = zeros(1,n);
normal3 = zeros(1,n);
binormal1 = zeros(1,n);
binormal2 = ones(1,n);
binormal3 = zeros(1,n);
for i = round(28/nvoxZ*n):1:n
    normal1(1,i) = cos((i-round(28/nvoxZ*n))*pi/(n-round(28/nvoxZ*n)));
    normal3(1,i) = sin((i-round(28/nvoxZ*n))*pi/(n-round(28/nvoxZ*n)));
end
normal = [normal1;normal2;normal3];
binormal = [binormal1;binormal2;binormal3];
% Initialization of node positions
xn = zeros(n,ncirc);
yn = zeros(n,ncirc);
zn = zeros(n,ncirc);
% Polar angle for nodes around circumference of cylinder cross-sections
s = linspace(2*pi/ncirc,2*pi,ncirc);
for k = 1:1:n
    for i = 1:1:ncirc
        xn(k,i) = XYZ(1,k) + initradius*(cos(s(1,i))*normal(1,k) + sin(s(1,i))*binormal(1,k));
        yn(k,i) = XYZ(2,k) + initradius*(cos(s(1,i))*normal(2,k) + sin(s(1,i))*binormal(2,k));
        zn(k,i) = XYZ(3,k) + initradius*(cos(s(1,i))*normal(3,k) + sin(s(1,i))*binormal(3,k));
    end
end
```

Figure 4.1: Pseudo-code (in MATLAB format) for initial cylindrical structured mesh generation.

To generate nodes around centerpoints, the centerline points were resampled and the local tangent, normal and binormal vectors to the centerline were evaluated at n points along the centerline. Nodes were created in n circular cross-sections perpendicular to the centerline, using a polar pattern:

$$\mathbf{X}^i = \mathbf{X}_0^i + \text{initradius} * (\cos(s) * \mathbf{N} + \sin(s) * \mathbf{B}) \quad (4.1)$$

where $\mathbf{N}$ and $\mathbf{B}$ are the local normal and binormal vectors to the centerline, respectively. $\mathbf{X}_0^i$ represent the position of a point on the centerline at cross-section i . Polar coordinate s places $ncirc$ nodes equally spaced along the circumference of the circle of radius $initradius$; therefore, $s \in [\frac{2\pi}{ncirc}, 2\pi]$. $\mathbf{X}^i$ is a 3-D node position with components (x_n, y_n, z_n) .

Figure 4.2 shows the centerline and a surface mesh for the synthetic aorta. The mesh is composed of 480 nodes, with 20 nodes along circumference and 24 cross-sections along the cylinder. The radius of the mesh is 8 mm.

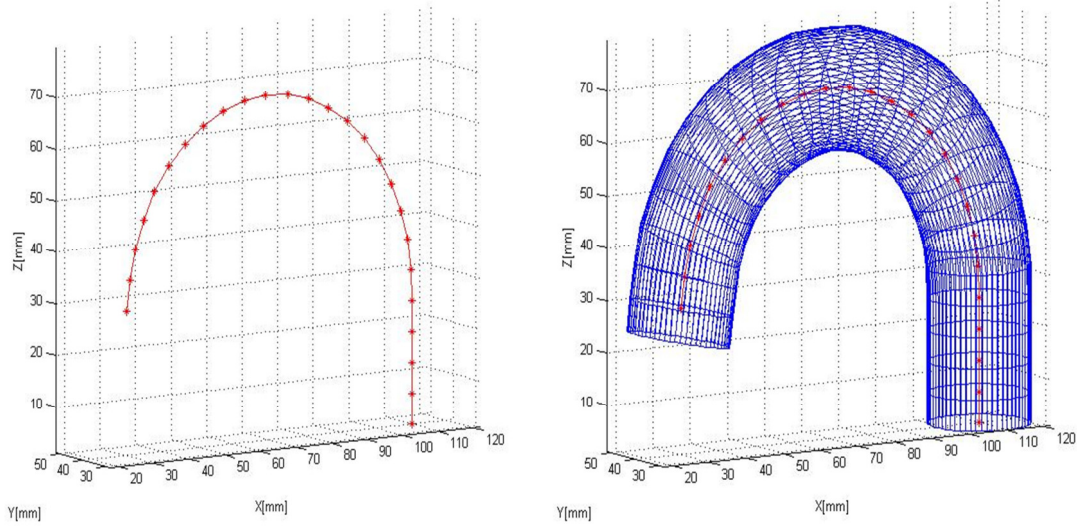


Figure 4.2: Centerline and arbitrary cylindrical structured mesh of the synthetic aorta.

4.2 Finite element formulation

In the context of a pressurized cylinder like the aorta, a surface mesh would normally be used in conjunction with shell finite elements. The formulation of such elements is well established in the literature, but was deemed too cumbersome for simply testing an idea. Instead, and because what only mattered was the creation of an aortic mesh and not the actual stress analysis of the aortic wall, the arbitrary cylindrical mesh was interpreted as a wireframe mesh made of 3-D beam finite elements, also known as space frame elements. Implementation was meant to take place in statics, where the equilibrium equations can be formulated as $\mathbf{K}\mathbf{d} = \mathbf{f}$, where $\mathbf{K}$ is the stiffness matrix for the cylinder, $\mathbf{d}$ is the vector of nodal degrees of freedom and $\mathbf{f}$ is the external load vector (stemming from the 3-D GVF field).

4.2.1 Derivation of the local stiffness matrix for a space frame element

A space frame element's loading and cross-sectional properties are described in a local $x - y - z$ coordinate system as shown in (Figure 4.3).

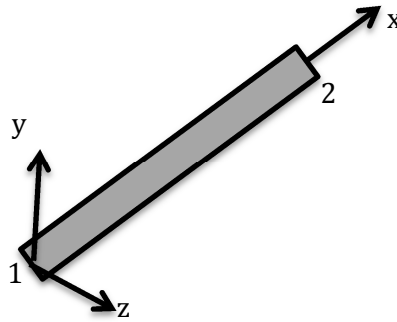


Figure 4.3: Space frame element, adapted from (Bhatti, 2006).

Each frame element consists of two nodes with 6 degrees of freedom each. In the local coordinate system, the degrees of freedom are (Figure 4.4):

d_1, d_2, d_3 Displacements at Node 1,

d_4, d_5, d_6 Rotations at Node 1,

d_7, d_8, d_9 Displacements at Node 2,

d_{10}, d_{11}, d_{12} Rotations at Node 2.

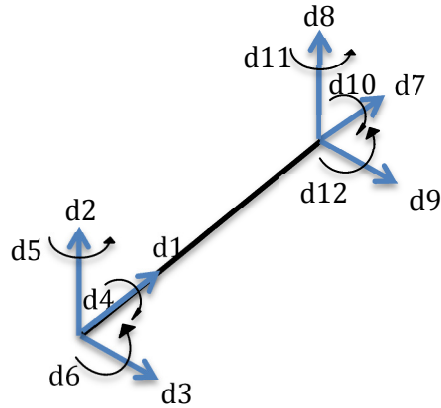


Figure 4.4: Local degrees of freedom for space frame element (Bhatti, 2006).

Following Bhatti, 2006, the axial displacements along $d_1 - d_7$ are interpolated as:

$$u(x) = \begin{bmatrix} 1 - \frac{x}{L} & \frac{x}{L} \end{bmatrix} \begin{pmatrix} d_1 \\ d_7 \end{pmatrix} \quad (4.3)$$

where $0 \leq x \leq L$ and L is the length of the element.

The displacement $v(x)$ in the y -direction is:

$$v(x) = \begin{bmatrix} 1 - \frac{3x^2}{L^2} + 2\frac{x^3}{L^3} & x - \frac{2x^2}{L} + \frac{x^3}{L^2} & \frac{3x^2}{L^2} - 2\frac{x^3}{L^3} & -\frac{x^2}{L} + \frac{x^3}{L^2} \end{bmatrix} \begin{pmatrix} d_2 \\ d_6 \\ d_8 \\ d_{12} \end{pmatrix} \quad (4.4)$$

Causing the bending in the y -direction are the moment about the z -axis and the shear force in the y -direction:

$$M_z(x) = EI_z \left(\frac{d^2 v}{dx^2} \right) \quad ; \quad V_y(x) = \frac{dM_z}{dx} = EI_z \left(\frac{d^3 v}{dx^3} \right)$$

Similarly, the displacement $w(x)$ in the z -direction is:

$$w(x) =$$

$$\begin{bmatrix} 1 - \frac{3x^2}{L^2} + 2\frac{x^3}{L^3} & -\left(x - \frac{2x^2}{L} + \frac{x^3}{L^2}\right) & \frac{3x^2}{L^2} - 2\frac{x^3}{L^3} & -\left(-\frac{x^2}{L} + \frac{x^3}{L^2}\right) \end{bmatrix} \begin{pmatrix} d_3 \\ d_5 \\ d_9 \\ d_{11} \end{pmatrix} \quad (4.5)$$

Causing the bending in the z -direction are the moment about the y -axis and the shear force in the z -direction:

$$M_y(x) = -EI_y \left(\frac{d^2 w}{dx^2} \right) \quad ; \quad V_z(x) = -\frac{dM_y}{dx} = EI_y \left(\frac{d^3 w}{dx^3} \right)$$

The rotation (twist) of the section about its x -axis is:

$$\psi(x) = \begin{bmatrix} 1 - \frac{x}{L} & \frac{x}{L} \end{bmatrix} \begin{pmatrix} d_4 \\ d_{10} \end{pmatrix} \quad 0 \leq x \leq L$$

The corresponding twisting moment per unit length is:

$$M_x(x) = GJ \left(\frac{d\psi}{dx} \right)$$

By combining the axial and bending in $x - z$ plane, and bending in $x - y$ plane deformations the total strain energy in a frame element can classically be written as follows:

$$U =$$

$$\frac{1}{2} \int_0^L EA \left(\frac{du}{dx} \right)^2 dx + \frac{1}{2} \int_0^L EI_z \left(\frac{d^2v}{dx^2} \right)^2 dx + \frac{1}{2} \int_0^L EI_x \left(\frac{d^2w}{dx^2} \right)^2 dx + \frac{1}{2} \int_0^L GJ \left(\frac{d\psi}{dx} \right)^2 dx$$

$$U = \frac{1}{2} \int_0^L \begin{bmatrix} \frac{du}{dx} & \frac{d^2v}{dx^2} & \frac{d^2w}{dx^2} & \frac{d\psi}{dx} \end{bmatrix} \begin{pmatrix} EA & 0 & 0 & 0 \\ 0 & EI_z & 0 & 0 \\ 0 & 0 & EI_x & 0 \\ 0 & 0 & 0 & GJ \end{pmatrix} \begin{pmatrix} \frac{du}{dx} \\ \frac{d^2v}{dx^2} \\ \frac{d^2w}{dx^2} \\ \frac{d\psi}{dx} \end{pmatrix} dx \quad (4.6)$$

Using the different interpolations given above,

$$\begin{pmatrix} \frac{du}{dx} \\ \frac{d^2v}{dx^2} \\ \frac{d^2w}{dx^2} \\ \frac{d\psi}{dx} \end{pmatrix} = \begin{pmatrix} -\frac{1}{L} & 0 & 0 & 0 & \dots \\ 0 & -\frac{6}{L^2} + \frac{12x}{L^3} & 0 & 0 & \dots \\ 0 & 0 & -\frac{6}{L^2} + \frac{12x}{L^3} & 0 & \dots \\ 0 & 0 & 0 & -\frac{1}{L} & \dots \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \\ \cdot \\ \cdot \\ d_{12} \end{pmatrix} = \mathbf{B}^T \mathbf{d}$$

where

B^T

$$= \begin{pmatrix} -\frac{1}{L} & 0 & 0 & 0 & 0 & 0 & \frac{1}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{12x}{L^3} - \frac{6}{L^2} & 0 & 0 & 0 & \frac{6x}{L^2} - \frac{4}{L} & 0 & \frac{6}{L^2} - \frac{12x}{L^3} & 0 & 0 & 0 & \frac{6x}{L^2} - \frac{2}{L} \\ 0 & 0 & \frac{12x}{L^3} - \frac{6}{L^2} & 0 & \frac{4}{L} - \frac{6x}{L^2} & 0 & 0 & 0 & \frac{6}{L^2} - \frac{12x}{L^3} & 0 & \frac{2}{L} - \frac{6x}{L^2} & 0 \\ 0 & 0 & 0 & -\frac{1}{L} & 0 & 0 & 0 & 0 & 0 & \frac{1}{L} & 0 & 0 \end{pmatrix}$$

Finally, the strain energy can be written in compact form as follows:

$$U = \frac{1}{2} \mathbf{d}^T \int_0^L \mathbf{B} \mathbf{C} \mathbf{B}^T dt \mathbf{d} = \frac{1}{2} \mathbf{d}^T \mathbf{K} \mathbf{d} \quad (4.7)$$

where

$$\mathbf{C} = \begin{pmatrix} EA & 0 & 0 & 0 \\ 0 & EI_z & 0 & 0 \\ 0 & 0 & EI_y & 0 \\ 0 & 0 & 0 & GJ \end{pmatrix}$$

contains the elastic properties as well as the cross-sectional and rigidity properties of the frame element. In the numerical experiments to follow, these values were arbitrarily set as shown in Table 4.1. Again, the goal was not to reproduce the mechanical properties of the real aorta, but simply to obtain a mesh that could deform under the influence of the 3-D GVF field.

Table 4.1 Material and circular cross-section properties for frame elements

Property	Value
E (Young's modulus, MPa)	10^3
ν (Poisson's ratio)	0.3
G (Shear modulus, MPa)	$\frac{E}{2 * (1 + \nu)}$
$I_z = I_y$ (mm ⁴)	$\frac{\pi/64}{2 * r^4}$
J (mm ⁴)	$2 * I_z$

Finally, the stiffness matrix in its local coordinates becomes:

$K=$

$$\begin{pmatrix} \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} \\ 0 & 0 & \frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 & 0 & 0 & -\frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 \\ 0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GJ}{L} & 0 & 0 \\ 0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 \\ 0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} \\ -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} & 0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} \\ 0 & 0 & -\frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 & 0 & 0 & \frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 \\ 0 & 0 & 0 & -\frac{GJ}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GJ}{L} & 0 & 0 \\ 0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 \\ 0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} \end{pmatrix}$$

Individual local frame element stiffness matrices are transformed into global ones using rotation matrices so that they refer to global degrees of freedom. Then, the global stiffness matrices are put together in a process called assemblage so as to represent the whole structure. The details for these steps are classic (see e.g. Bhatti, 2006), present little interest for the reader in regards to the present work, and therefore are omitted.

The boundary conditions associated with the synthetic aortic model were defined as Dirichlet boundary conditions at two locations. More specifically, the nodes in the first and last cross-sections of the cylinder were fixed in all directions. This was necessary to prevent rigid body motions that would make the assemblage stiffness matrix singular.

4.2.2 Using the 3-D GVF field as a force field

In this approach, it is proposed to use the arbitrary mesh of the cylindrical aortic shape as a structural FE model to be deformed by the forces of the 3-D GVF field. The external loads applied to all n nodes of the mesh are defined as:

$$\mathbf{F}^i = \rho \mathbf{U}^i \quad i = 1, \dots, n \quad (4-8)$$

where ρ is a constant arbitrary coefficient used to scale the magnitudes of the GVF vectors $\mathbf{U}^i$ which are made of the components of u, v and w of the GVF field.

After one iteration, the nodes move to places where the GVF vectors are different. Therefore, the components of the GVF are updated at the end of each iteration, until a sign change in the components of the GVF vectors is detected. A sign change in the components of the GVF vectors means that a boundary has been crossed; therefore, deformation of the mesh should stop.

4.2.3 Results

Figure 4.5 illustrates the deformed mesh after 36 iterations, with $\rho = 500$. It took about an hour to produce these results. The iterations were arbitrarily stopped to view the process of mesh expansion. As can be seen, the smoothness and continuity of the initial mesh is preserved.

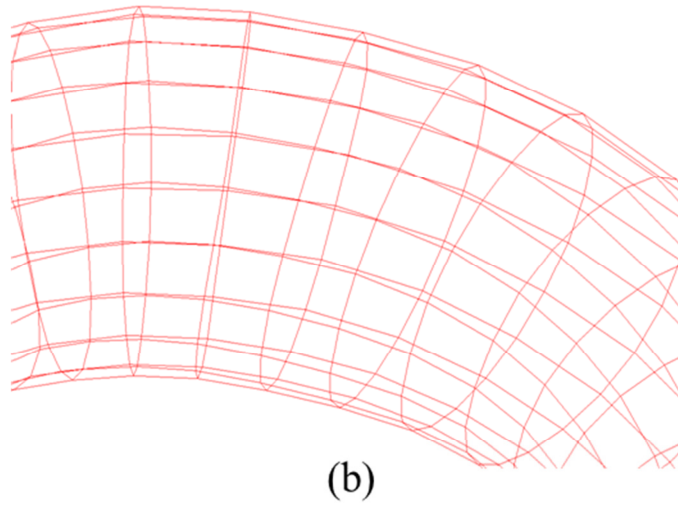
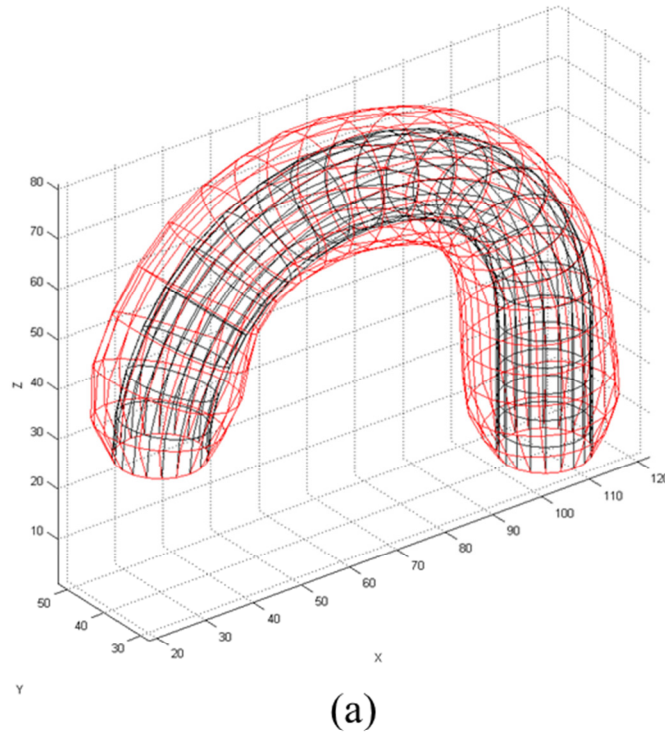


Figure 4.5: (a) Initial mesh (black), and deformed mesh (red) using 3-D GVF field as applied force, after 36 iterations, (b): Magnified deformed mesh.

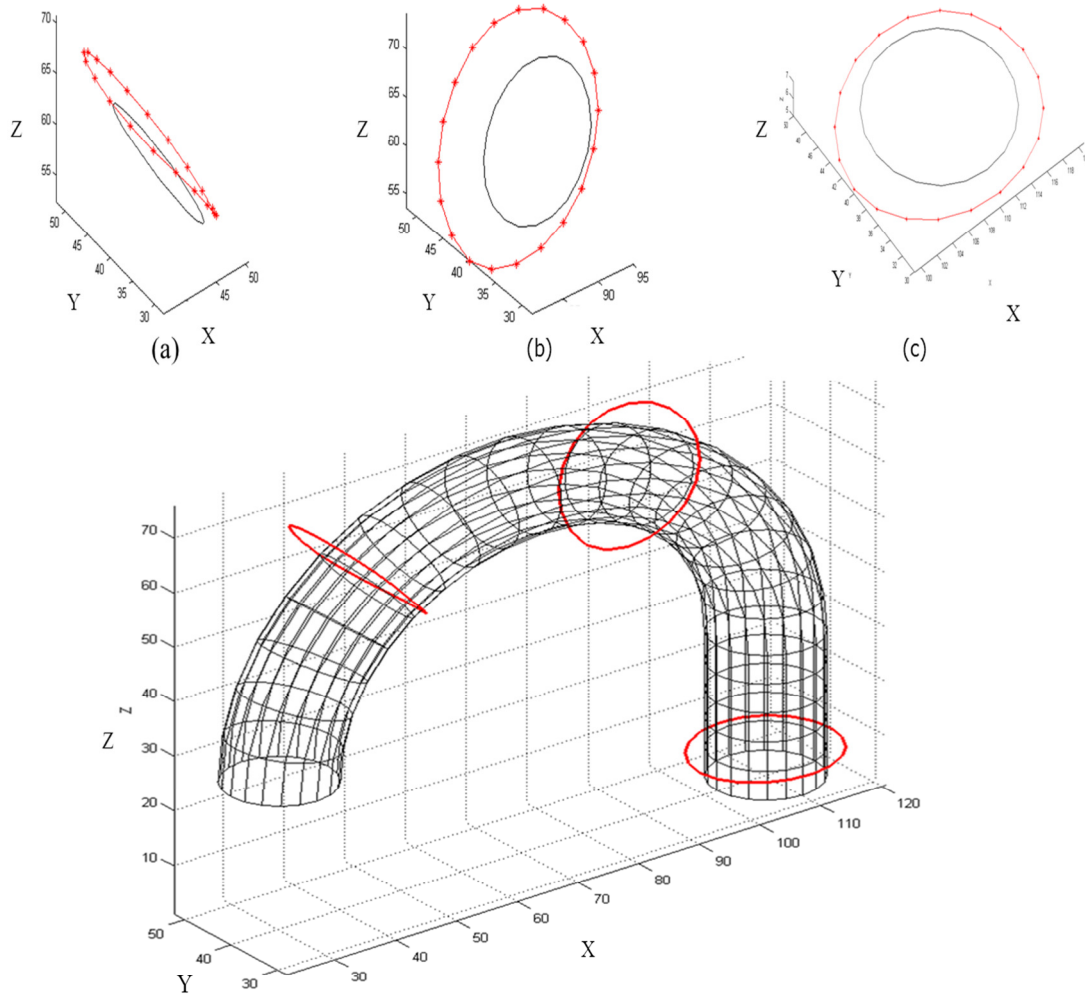


Figure 4.6: Location of original initial nodes (black) and deformed nodes (red), at (a): Cross section #19, (b): Cross section #12 and (c): Cross section #2, after 36 iterations.

Figure 4.6 highlights the fact that some cross-sections of the mesh get more deformation than others. Figure 4.7 shows the ultimate consequence of this issue when the iterations were let run for more iterations,

It took several hours and 138 iterations for the mesh expansion to stop, not due to strict satisfaction of the convergence criterion, but because the mesh was going out of the 3-D domain of the original image. Figure 4.7 and Figure 4.8 provide more details on this issue.

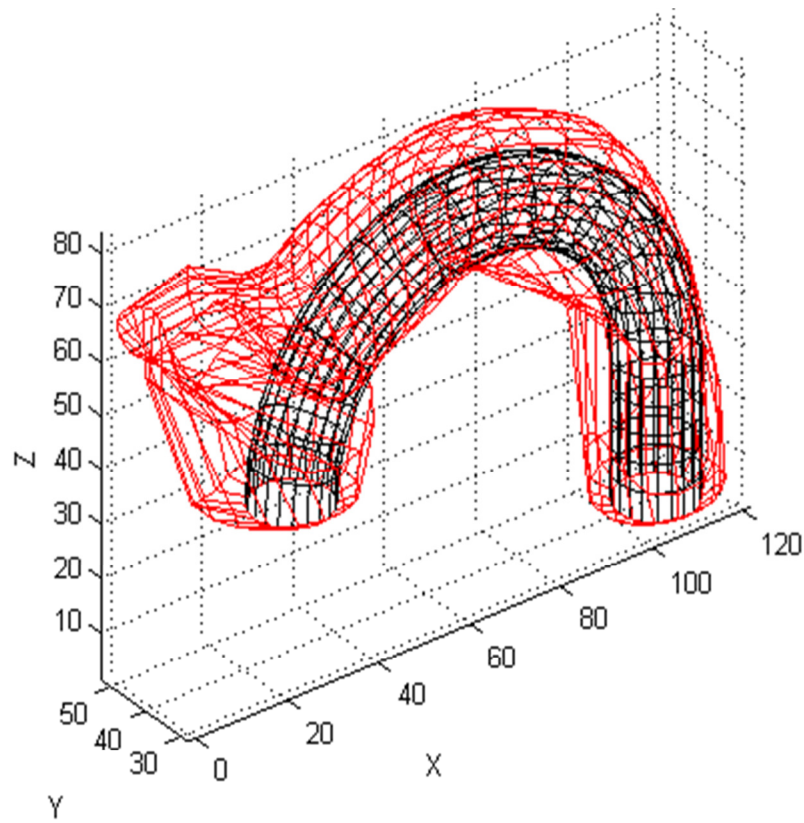


Figure 4.7: Initial mesh (black), and deformed mesh (red) using 3-D GVF field as applied force after 138 iterations.

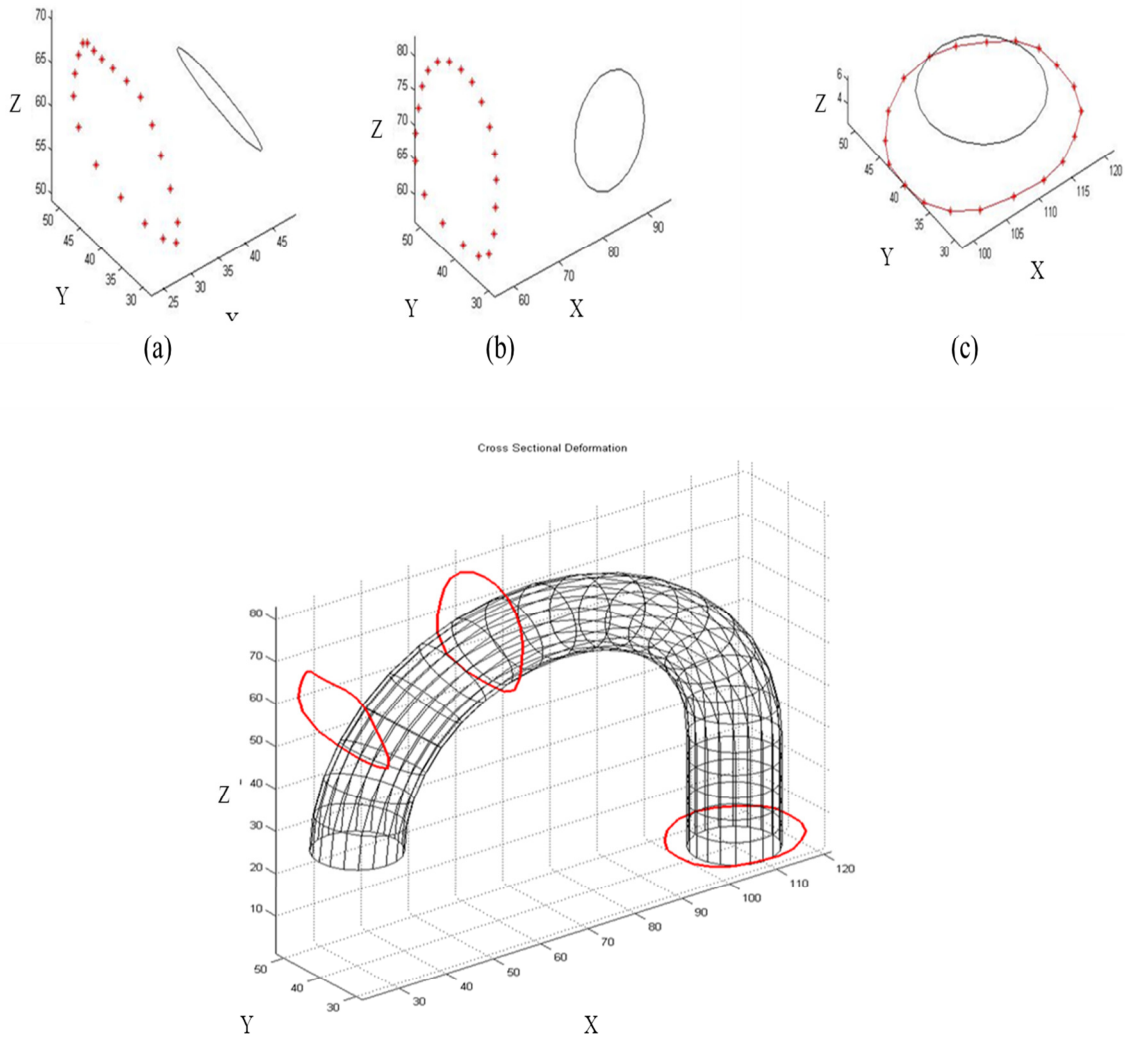


Figure 4.8: Locations of original initial nodes (black) and deformed nodes (red), at (a): Cross section #19, (b): Cross section #12 and (c): Cross section #2.

4.2.4 Discussion

Application of 3-D GVF field vectors interpreted as forces acting on a real 3-D structural mesh was not successful, although smooth deformations were first observed. Several issues arose that we were not able to resolve. One was processing time. Each iteration required a new assemblage stiffness matrix to be put together and inverted for solution.

This came at a significant processing cost and, in the end, turned out to be impractical, although dedicated solvers could have been used if other characteristics of the method had been more promising. Another issue was the difficulty in determining an efficient stopping criterion. Many were tried, other than the one described, but none was successful. No overall improvement was observed when varying the scaling parameter ρ . Smaller values would reduce the incremental growth of the mesh, but would not improve convergence. Similar effects were noted when changing the material properties of the rods of the wireframe mesh. In brief, in spite of all the simplifications introduced in the approach, it was never possible to have the initial mesh to conform to the aortic shape of interest. This led us to try a different approach, wherein the GVF field would be interpreted as imposed displacement vectors, as discussed next. Although inspired from the previous approach, this second approach did not rely on a finite element formulation.

4.3 Using the 3-D GVF as an imposed displacement field

Again, the 3-D GVF field obtained from the synthetic aortic shape was employed in an attempt to deform the same initial cylindrical mesh, although deprived of material properties, into the aortic shape itself. The components of the 3-D GVF were interpreted as displacements imposed to all the n nodes of the initial mesh, such that

$$\mathbf{X}^i = \mathbf{X}^0 + \rho \mathbf{U}^i \quad i = 1 \dots n \quad (4-9)$$

where ρ is a coefficient used to scale the magnitudes of the GVF vectors, $\mathbf{X}^0$ is the original position of node i , and $\mathbf{X}^i$ its new position. Again, a stopping criterion was

devised, based on the change in direction of vectors in the GVF field. The mesh deformation algorithm is shown in Figure 4.9.

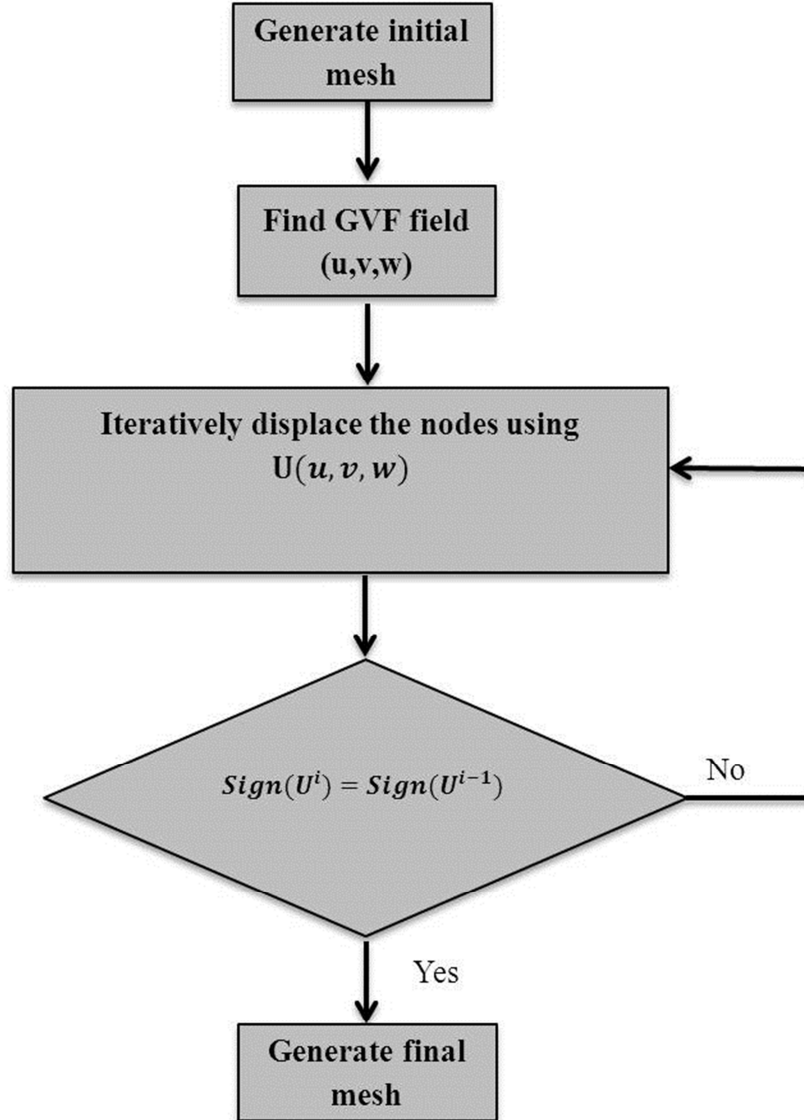
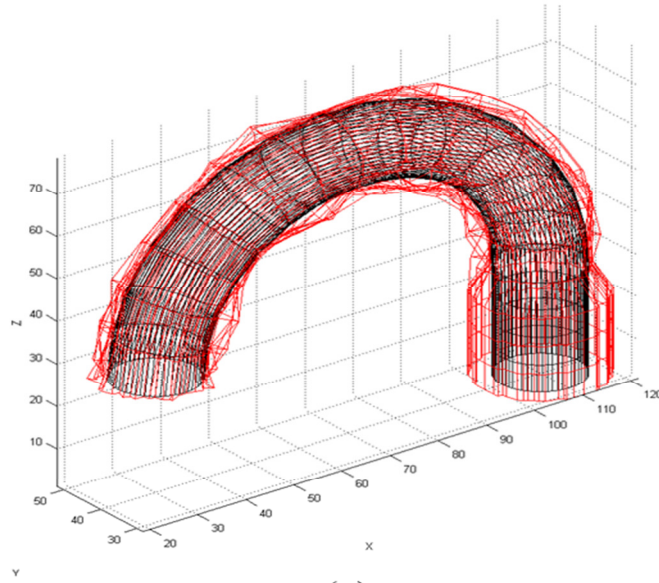


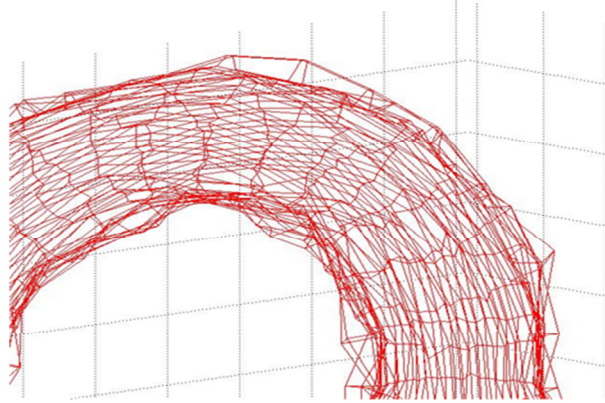
Figure 4.9: Mesh deformation algorithm based on 3-D GVF field interpreted as imposed displacement.

4.3.1 Results

Using the procedure described in Figure 4.1, an initial mesh was generated with 60 nodes in the circumference and 20 cross-sections in the longitudinal direction, for a total of 800 nodes. The initial mesh and that deformed under the influence of the 3-D GVF are shown in Figures 4.10 and 4.11.



(a)



(b)

Figure 4.10: a) Original mesh (black), and deformed mesh (red) using 3-D GVF field interpreted as imposed displacement field, b) magnified deformed mesh.

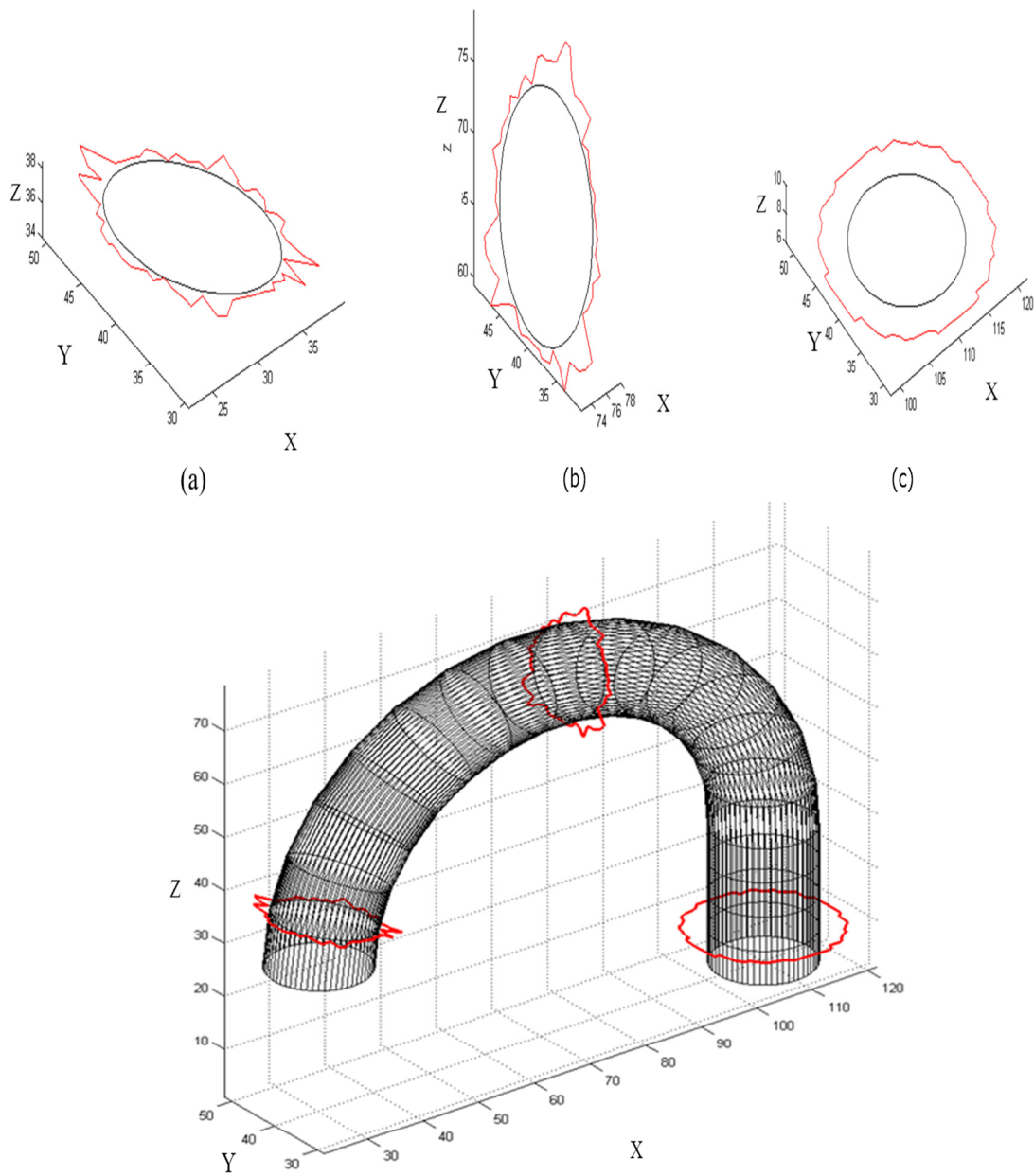


Figure 4.11: Locations of original initial nodes (black) and deformed nodes (red), at (a): Cross section #19, (b): Cross section #12 and (c): Cross section #2.

4.3.2 Discussion

The computation time was only on the order of a few minutes, and convergence was achieved. As can be seen from Figures 4.10 and 4.11 the imposed displacements produced a smooth mesh in the straight portion of the aortic model. However, in the curved portion of the aorta, the deformed mesh became very distorted.

Overall, although the 3-D GVF displayed features that were desirable toward the goal of this work, it appeared that none of the methods as presented herein to implement it on synthetic images was satisfactory, due to accuracy and convergence problems and/or slow processing speed. In addition, in these attempts, it was assumed that the centerline of the aorta was known a priori. This would not be the case in a real-life scenario, where determination of blood vessel centerline in medical images is an active field of research on its own. A successful scheme must be fast and robust, and must not require a priori knowledge of the centerline of the aorta. It must also feature high accuracy to be acceptable as a competitor in medical image analysis. To address these requirements, we were led to develop another approach from scratch, which yielded the new algorithms proposed in the next chapter.

5 Automatic surface mesh generation

The novel proposed methodology relies on the 3-D GVF field as an effective tool to deform an arbitrary structured mesh into a target aortic wall shape, thereby segmenting and meshing the aorta in one single effort. However, compared to the failed attempts presented in the previous chapter, wherein the arbitrary structured mesh was built first and then deformed as a whole, the new approach grows the arbitrary structured mesh on the fly, as the segmentation and meshing proceed from one end of the aorta to the other.

Before embarking on the description of the core features of the methodology, it is timely to clarify the notion of region of interest (ROI). In the case at hand, with synthetic or medical images containing information about the aorta, the user needs to specify the position and size of a 3-D box (the ROI) within which the aorta is to be analyzed and reconstructed. Use of an ROI reduces the search region for reconstruction and reduces processing time. The selection is carried out by cropping the original 3-D images to only keep the necessary objects within the limits of the ROI. A graphical-user interface (GUI) was created for the user to easily crop the original 3-D image along three perpendicular planes to identify the ROI in the x , y , and z directions. Once the ROI is chosen (Figure 5.1), it is saved for future processing as detailed in the next sections.

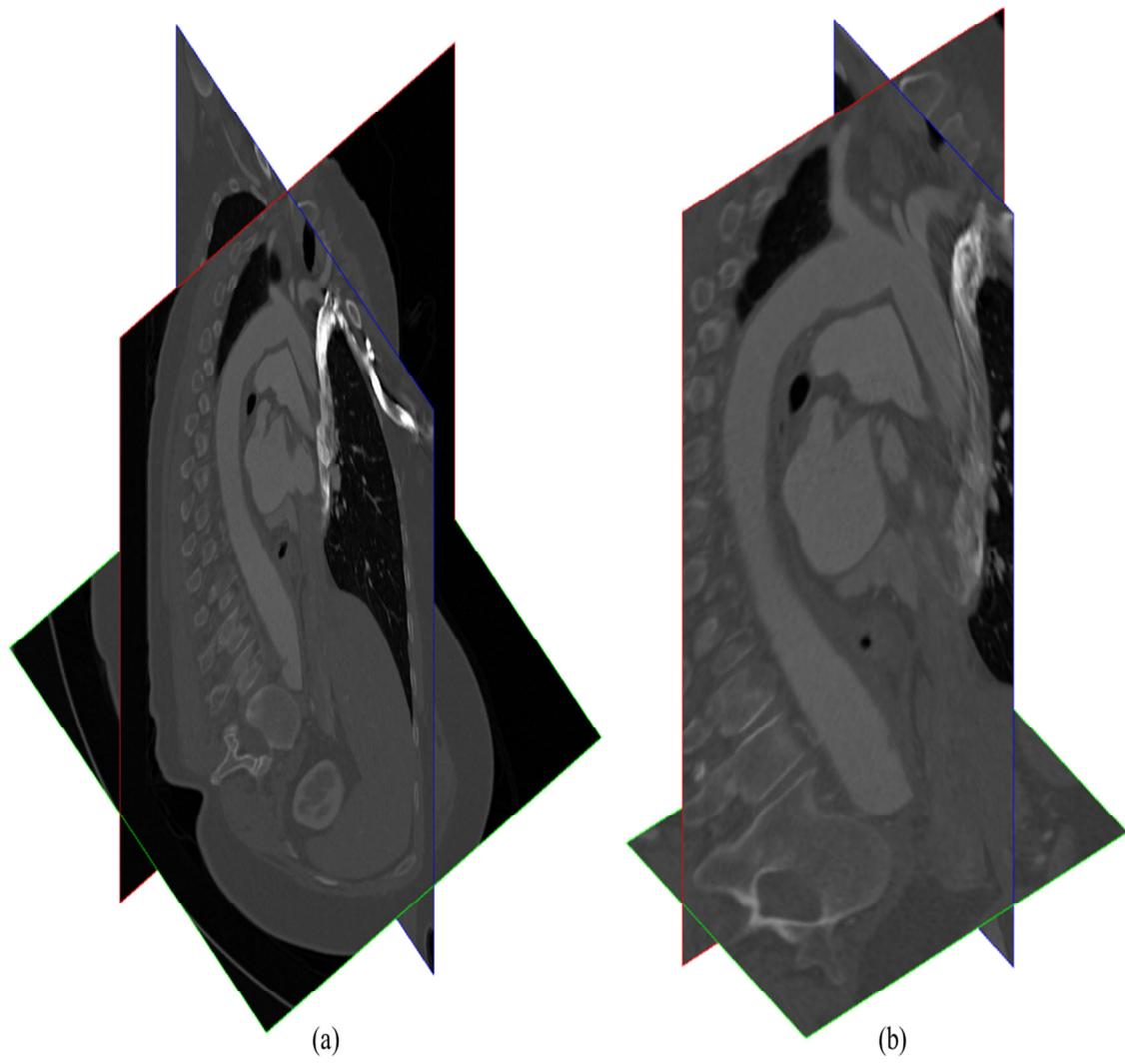


Figure 5.1: (a): Original image volume from a CT dataset, (b): Region of interest (ROI).

5.1 Methodology

The new methodology starts with the computation of the 3-D GVF field for the whole ROI. This computation is meant to be done only once for a given dataset. Depending on the size, resolution and level of noise of the datasets, the processing time differs.

However, it does not exceed a few minutes and requires no input from the user. Then, starting from a first approximate point and an approximate direction input by the user, a ray-tracing algorithm is used to find the aortic wall and locate the centroid of a local cross-section; a marching method is then used to repeat the process from one local cross-section to the next and generate a structured FE mesh. The preliminary structured mesh obtained in this fashion is then further refined according to the user's requirements.

The detailed methodology (Figure 5.2) is explained in the coming sections. The noiseless synthetic aortic dataset presented in Chapter 3 (Figure 3.1) and shown again for convenience in Figure 5.3 is used to illustrate the different steps followed by the method. Thereafter, four different real datasets will be processed as examples of applications.

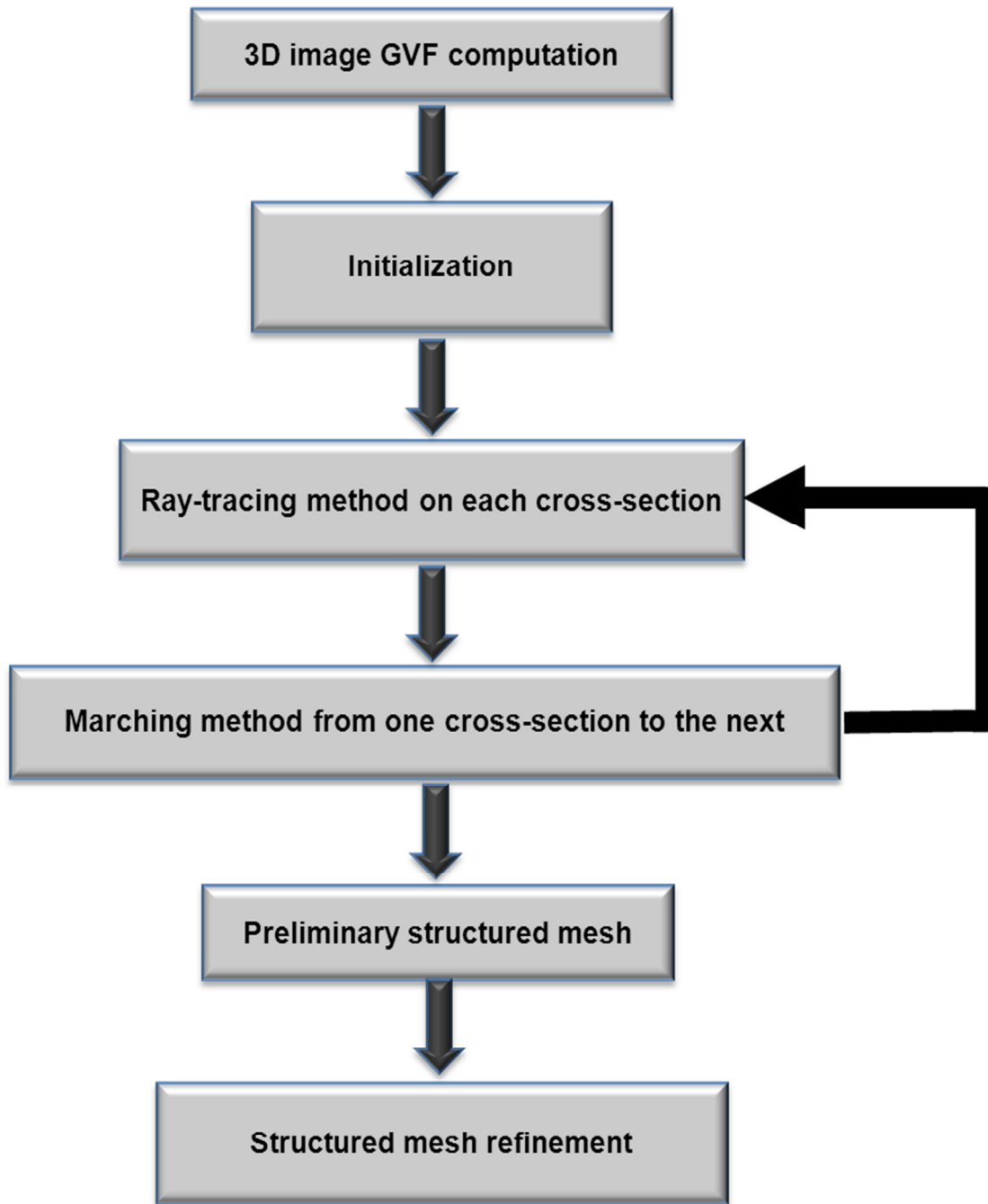


Figure 5.2: Surface mesh generation algorithm.

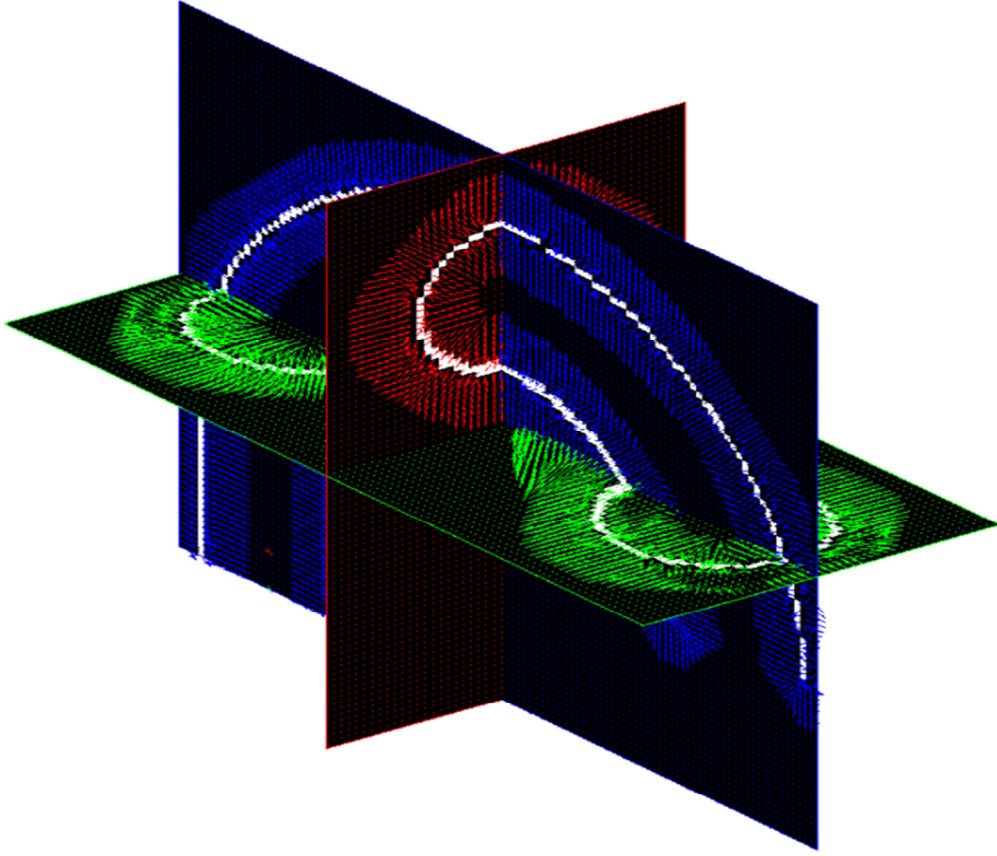


Figure 5.3: 3-D GVF field for synthetic aortic image obtained after 6 iterations and a few seconds of computation.

5.2 Initialization

Once the 3-D GVF field is computed, the only user input in the automatic mesh generation algorithm in Figure 5.2 is the initial point and the direction for mesh generation. Using a dedicated GUI, the user needs to approximate the center of one aortic cross-section in the image volume. To identify the direction for mesh generation, the user needs to approximate the center of a second aortic cross-section in the desired direction. This is done in the GUI using slider bars controlling the x , y and z locations of orthogonal planes and a selection button as illustrated in Figure 5.4.

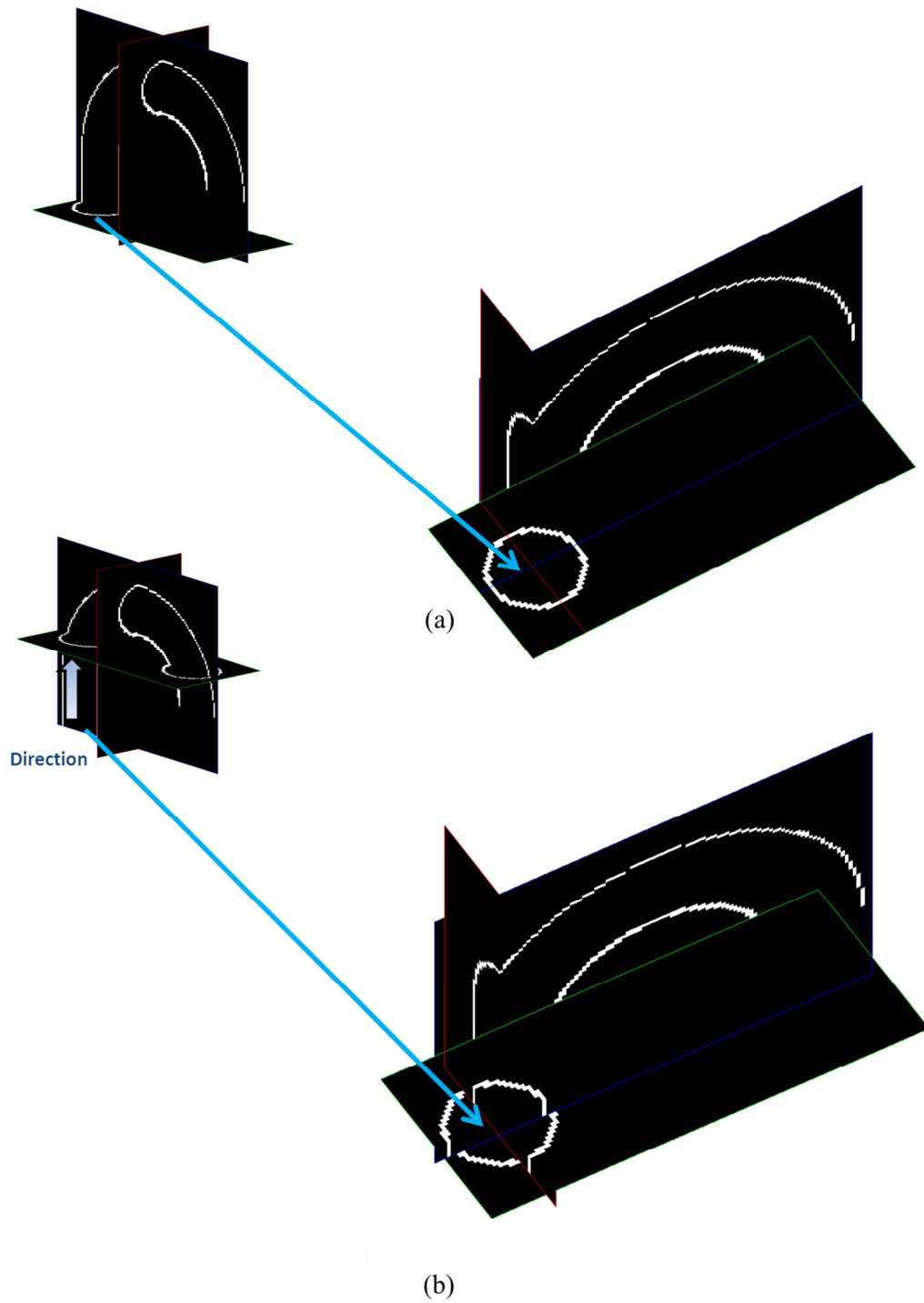


Figure 5.4: (a) Selection of first cross-section, (b) Selection of second cross-section to direction for mesh generation.

5.3 Ray-tracing method

After an approximate center point is defined in current cross-section k , a modified ray-tracing method is used to find the aortic wall and accurately locate the centroid of this cross-section. Ray-tracing is a classic image generation method that usually traces the path of light through a virtual 3-D environment and simulates the effects of its encounters with virtual objects. This method determines visible surfaces in an image at the pixel level (Appel, 1968).

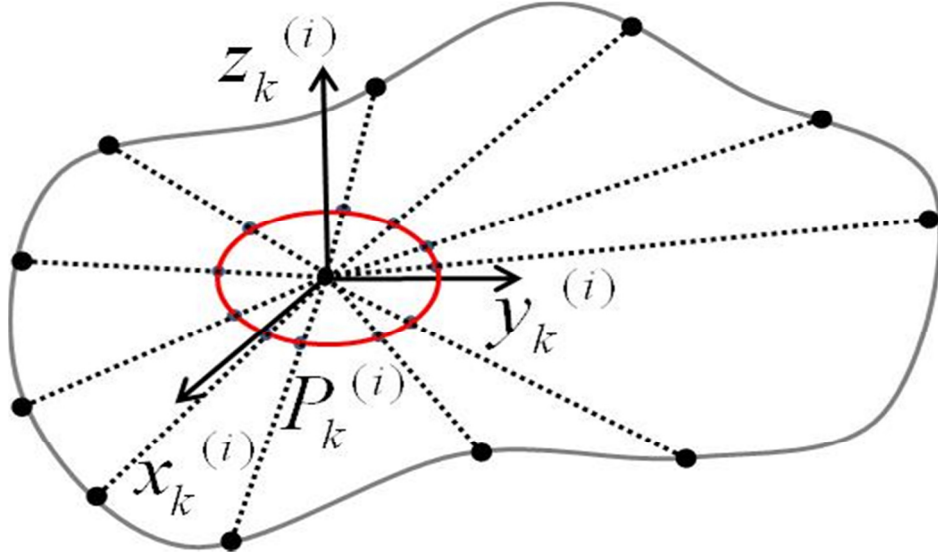


Figure 5.5: Partial illustration of the ray-tracing algorithm for cross-section k at iteration i . The 3-D GVF field is not represented for clarity.

We use the ray-tracing concept to exploit the properties of the 3-D GVF field. As shown in Figure 5.5., in aortic cross-section k at iteration i , the approximate center $P_k^{(i)}$ and associated coordinate system $(x_k^{(i)}, y_k^{(i)}, z_k^{(i)})$ are used to create $ncirc$ equally spaced nodes $(x_{nk}(j), y_{nk}(j), z_{nk}(j))$, $j \in [1, ncirc]$ around a small circle inside the aorta. The radius of this initial circle (shown in red in Figure 5.5) is arbitrary. It simply needs to be

small enough so that the initial circle lies inside the aorta. For each of the initial nodes on the circle, the radius is then individually increased through an iterative process until the aortic wall is reached. The boundary detection takes advantage of the properties of the 3-D GVF field, whereby its vectors change directions at boundaries. Specifically, if the dot product of a local 3-D GVF vector and the local radius vector becomes negative, this means that the node has reached the aortic wall. If, because of noise or special circumstances, the algorithm does not find the aortic wall, the radius for one node or several nodes may theoretically become infinite. This possibility is eliminated by checking that the nodes stay within the boundaries of the ROI, and that the radius does not become unduly large. For each node, the last position found in this manner inside the aorta is chosen as belonging to the boundary of cross-section k .

Then, the node positions within the cross-section are smoothed using a classical central moving average method. Finally, centroid $P_k^{(i+1)}$ is determined as:

$$P_k^{(i+1)} = \frac{1}{ncirc} \sum_{j=1}^{ncirc} (x_{nk}(j), y_{nk}(j), z_{nk}(j)) \quad (5.1)$$

where $ncirc$ is the number of nodes along the circumference of the cross-section.

Convergence of the iterations for cross-section k is reached when $P_k^{(i+1)}$ and $P_k^{(i)}$ become the same point P_k , within a set precision.

The ray-tracing algorithm is presented in Figure 5.6.

Input: Number of nodes requested on each cross-section (ncirc), arbitrary initial radius for aortic mesh (radius), approximate coordinates of centroid (Center), 3-D GVF field components (u, v, w), precision (eps)

Output: centroid of cross-section (New_center), actual local aortic radius (finalradius)

```
s = linspace(2*pi/ncirc,2*pi,ncirc);
spangenerator = 2; % span of moving average = 2*spangenerator + 1
counter = 1; New_center = Center; New_radius = zeros(ncirc,1);
while (norm(Center - New_center) > eps || counter == 1) && counter < 10
    Center = New_center;
    for i = 1:ncirc
        out_of_bounds_test = 0;
        radius(i,1) = initradius;
% find nodal coordinates using the radius information (next three lines):
        xn(1,i)=Center(1,1)+radius(i,1)*(cos(s(1,i))*xx(1,1)+sin(s(1,i))*yy(1,1));
        yn(1,i)=Center(2,1)+radius(i,1)*(cos(s(1,i))*xx(2,1)+sin(s(1,i))*yy(2,1));
        zn(1,i)=Center(3,1)+radius(i,1)*(cos(s(1,i))*xx(3,1)+sin(s(1,i))*yy(3,1));
        if round(xn(1,i)/calX) < 1 || round(xn(1,i)/calX) > nvoxX
            out_of_bounds_test = 1;
        elseif round(yn(1,i)/calY) < 1 || round(yn(1,i)/calY) > nvoxY
            out_of_bounds_test = 1;
        elseif round(zn(1,i)/calZ) < 1 || round(zn(1,i)/calZ) > nvoxZ
            out_of_bounds_test = 1;
        end
        while out_of_bounds_test == 1
            out_of_bounds_test = 0;
            radius(i,1) = radius(i,1) - eps;
            if radius(i,1) > 0
            else
                radius(i,1) = 0;
            end
        end
% find nodal coordinates using the radius information
        if round(xn(1,i)/calX) < 1 || round(xn(1,i)/calX) > nvoxX
            out_of_bounds_test = 1;
        elseif round(yn(1,i)/calY) < 1 || round(yn(1,i)/calY) > nvoxY
            out_of_bounds_test = 1;
        elseif round(zn(1,i)/calZ) < 1 || round(zn(1,i)/calZ) > nvoxZ
            out_of_bounds_test = 1;
        end
    end
    if out_of_bounds_test == 0
        F(1,1) =v(round(xn(1,i)/calX),round(yn(1,i)/calY),round(zn(1,i)/calZ));
        F(2,1) =u(round(xn(1,i)/calX),round(yn(1,i)/calY),round(zn(1,i)/calZ));
        F(3,1) =w(round(xn(1,i)/calX),round(yn(1,i)/calY),round(zn(1,i)/calZ));
        ray = [xn(1,i) -Center(1,1);yn(1,i) - Center(2,1);zn(1,i)-Center(3,1)];
        out_of_bounds_test2 = 0;
        while dot(ray,F) >= 0 && out_of_bounds_test2 == 0
            if radius(i,1) < 20*initradius
                radius(i,1) = radius(i,1) + eps;
            end
        end
% find nodal coordinates using the radius information
```

```

% check whether the calculated nodes are within the GVF boundaries
    if round(xn(1,i)/calX) < 1 || round(xn(1,i)/calX) > nvoxX
        out_of_bounds_test2 = 1;
    elseif round(yn(1,i)/calY) < 1 || round(yn(1,i)/calY) > nvoxY
        out_of_bounds_test2 = 1;
    elseif round(zn(1,i)/calZ) < 1 || round(zn(1,i)/calZ) > nvoxZ
        out_of_bounds_test2 = 1;
    end
% For nodes that are out of GVF boundaries
    if out_of_bounds_test2 == 1
        radius(i,1) = radius(i,1) - eps;
        if radius(i,1) > 0
            else
                radius(i,1) = 0;
            end
    end
% find nodal coordinates using the radius information
    else
        F(1,1) = v(round(xn(1,i)/calX),round(yn(1,i)/calY),round(zn(1,i)/calZ));
        F(2,1) = u(round(xn(1,i)/calX),round(yn(1,i)/calY),round(zn(1,i)/calZ));
        F(3,1) = w(round(xn(1,i)/calX),round(yn(1,i)/calY),round(zn(1,i)/calZ));
        ray = [xn(1,i) - Center(1,1);yn(1,i) - Center(2,1);zn(1,i) - Center(3,1)];
    end
    end
else
    xn(1,i) = Center(1,1);
    yn(1,i) = Center(2,1);
    zn(1,i) = Center(3,1);
end
end
newradius = zeros(2*spangenerator + ncirc,1);
for l = 1:spangenerator
    newradius(l,1) = radius(ncirc - spangenerator + l,1);
    newradius(spangenerator + ncirc + l,1) = radius(l,1);
end
for i = 1:ncirc
    newradius(spangenerator + i,1) = radius(i,1);
end
    finalradius = smooth(newradius,2*spangenerator + 1);
for i = 1:ncirc
    New_radius(i,1) = finalradius(spangenerator + i,1);
% find nodal coordinates using the radius information
end
New_center(1,1) = sum(xn(1,:),2)/ncirc;
New_center(2,1) = sum(yn(1,:),2)/ncirc;
New_center(3,1) = sum(zn(1,:),2)/ncirc;
counter = counter + 1;
end

```

Figure 5.6: Abridged ray-tracing algorithm (in MATLAB format).

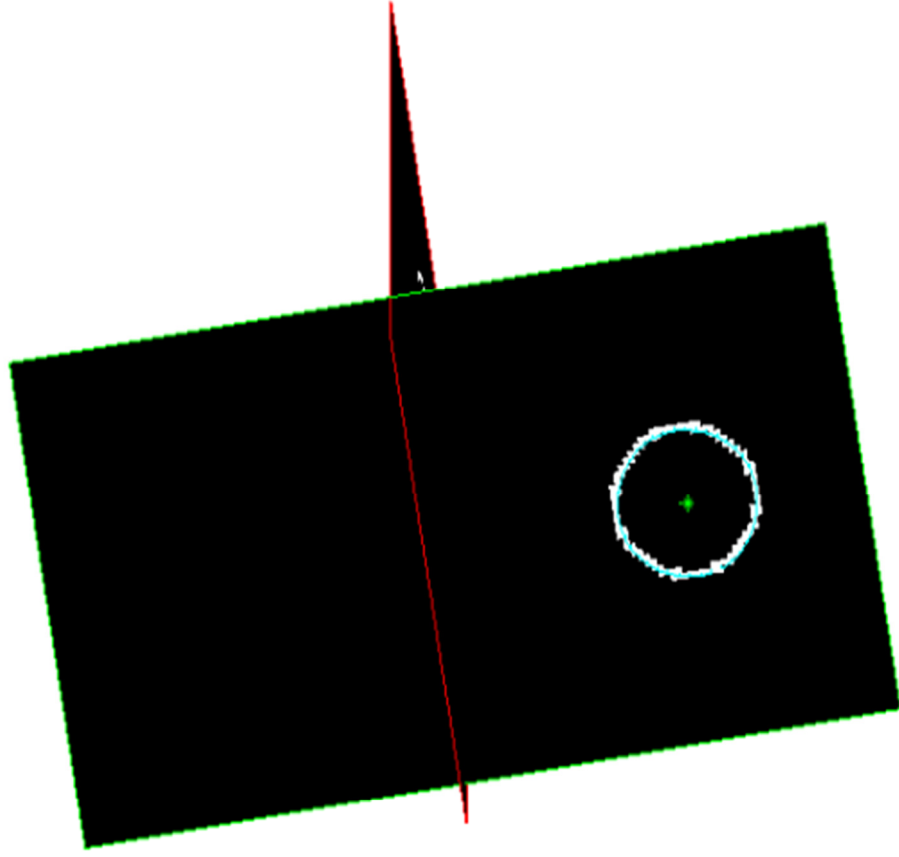


Figure 5.7: Ray-tracing method application on cross-section k .

Figure 5.7 presents the application of the ray-tracing method on cross-section k . The white circle is the wall of the synthetic aorta and the light blue circle is the detected inner boundary. The green point in the center of the circle is the detected centroid.

5.4 Marching method

As previously explained, the general direction of displacement from one cross-section to the next was defined during initialization. From the finalized cross-section k , the location and orientation of a tentative cross-section $k + 1$ are created by moving P_k

by some arbitrary distance along z_k , thereby defining $P_{k+1}^{(1)}$ and associated coordinate system $(x_{k+1}^{(1)}, y_{k+1}^{(1)}, z_{k+1}^{(1)})$ at iteration 1 (Figure 5.8). The ray-tracing algorithm is then used to determine a more accurate center of gravity $P_{k+1}^{(2)}$ for cross-section $k + 1$. The local coordinate system is updated to $(x_{k+1}^{(2)}, y_{k+1}^{(2)}, z_{k+1}^{(2)})$ where $z_{k+1}^{(2)}$ is along direction $P_k P_{k+1}^{(2)}$ and appropriate rotation angles are identified to determine $x_{k+1}^{(2)}$ and $y_{k+1}^{(2)}$ (Figure 5.9). Convergence of the iterations for cross-section $k + 1$ is reached when $P_{k+1}^{(i+1)}$ and $P_{k+1}^{(i)}$ become the same point P_{k+1} , within a set precision.

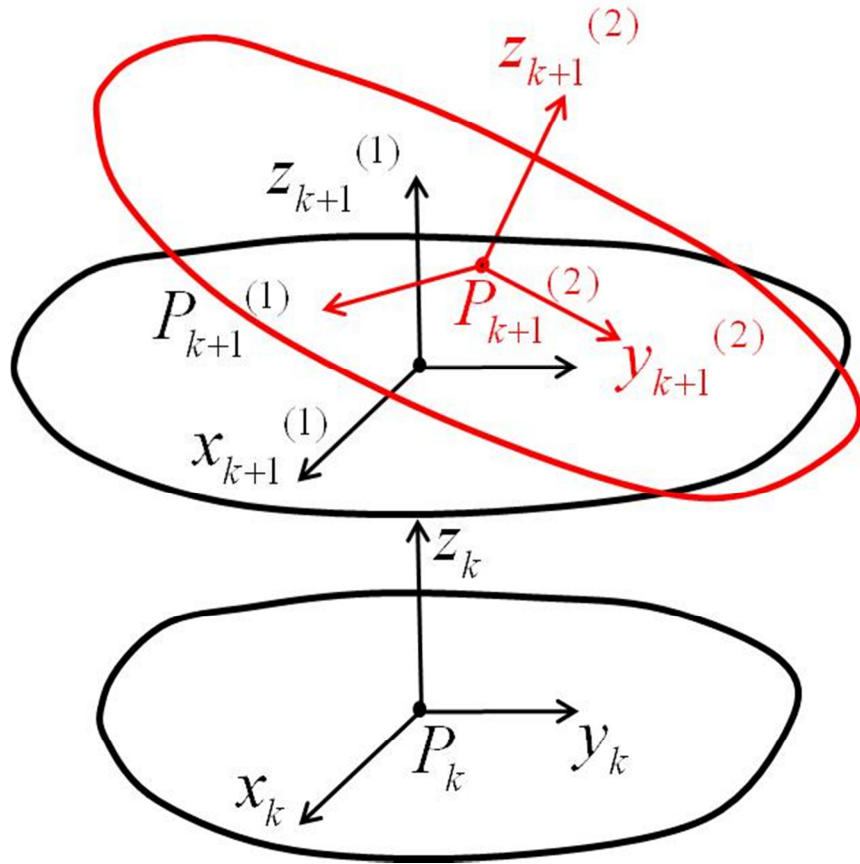


Figure 5.8: Partial illustration of the marching algorithm to image and mesh cross-section $k + 1$ starting from cross-section k . The 3-D GVF field is not represented for clarity.

The rotation angles are determined as follows. At iteration 1, the local coordinate system $(x_{k+1}, y_{k+1}, z_{k+1})$ is known and depending on the Δz (between two centroids of consecutive cross-sections) the rotation angles β and θ must be calculated (Figure 5.9) to find the updated coordinate system $(x_{k+1}^{i+1}, y_{k+1}^{i+1}, z_{k+1}^{i+1})$.

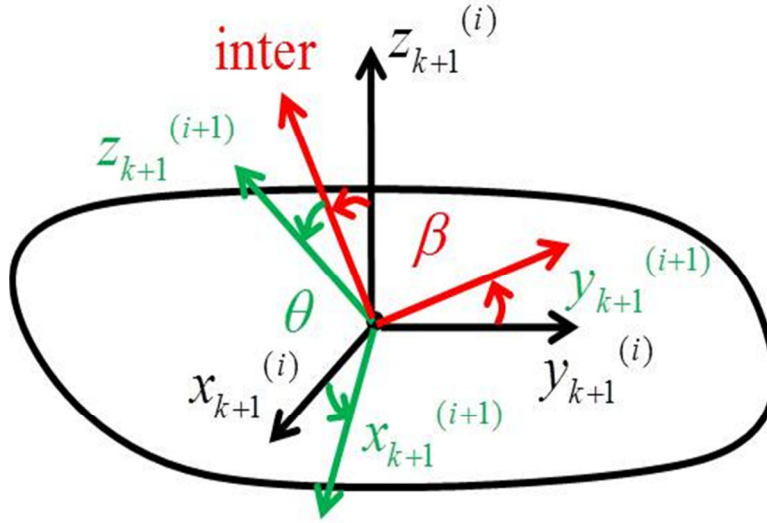


Figure 5.9: Coordinate system and rotation angles

As can be seen from Figure 5.9, the intermediate vector ($Inter_{k+1}$) is defined such that

$Inter_{k+1} = \cos(\beta) z_{k+1}^i - \sin(\beta) y_{k+1}^i$. Similarly,

$$y_{k+1}^{i+1} = \cos(\beta) y_{k+1}^i + \sin(\beta) z_{k+1}^i, \quad (5.2)$$

$z_{k+1}^{i+1} = \cos(\theta) Inter_{k+1} + \sin(\theta) x_{k+1}^i$, and

$$x_{k+1}^{i+1} = \cos(\theta) x_{k+1}^i - \sin(\theta) Inter_{k+1}. \quad (5.3)$$

Using Equations (5.2) and (5.3),

$$z_{k+1}^{i+1} = \sin(\theta) x_{k+1}^i - \cos(\theta) \cos(\beta) y_{k+1}^i - \cos(\theta) \cos(\beta) z_{k+1}^i$$

From this equation, one can extract the rotation angles as

$$\theta = \arcsin(z_{k+1}^{i+1} \cdot x_{k+1}^i) \quad (5.4)$$

and

$$\beta = -\arcsin\left(\frac{z_{k+1}^{i+1} \cdot y_{k+1}^i}{\cos(\theta)}\right).$$

Once the rotation angles are found, the new coordinate system for the next cross-section is calculated. Consequently, using the marching scheme, we can apply the ray-tracing algorithm to find the next cross-section's boundary centroid and orientation. The same routine carries on as long as the user keeps on hitting a button in the GUI. Note that the default arbitrary initial radius of the mesh and the arbitrary Δz representing the distance between two cross-sections can be modified at any time by the user if necessary.

The basic scheme outlined above worked well for straight portions of the aorta, but needed refinement for curved portions, especially in noisy datasets or around bifurcations of the aorta. The specific issue was that the rotation angles from one cross-section to the next should not vary widely if one wants to preserve a smooth centerline for the aorta. Therefore, an orientation control algorithm was devised, as detailed next.

5.5 Orientation control algorithm

The orientation control procedure implements limits on the tortuosity of the reconstructed centerline. Practically, if one does not want two consecutive cross-sections to intersect, there should be maximal values that the rotation angles should not exceed. In other words, the maximum allowable rotations were expressed as

$$f(\tau, \Delta z, r) = \sin^{-1}\left(\tau * \frac{\Delta z}{0.5 * (\max(r^{k-1}) + \max(r^k))}\right) \quad (5.5)$$

where Δz is the distance between two centroids of consecutive cross-sections, and r^k and r^{k-1} are the cross-sectional radii at iterations k and $k - 1$ respectively. As illustrated in Figure 5.12, τ is a proportion (between 0 and 1) of Δz allowed as maximum rotation.

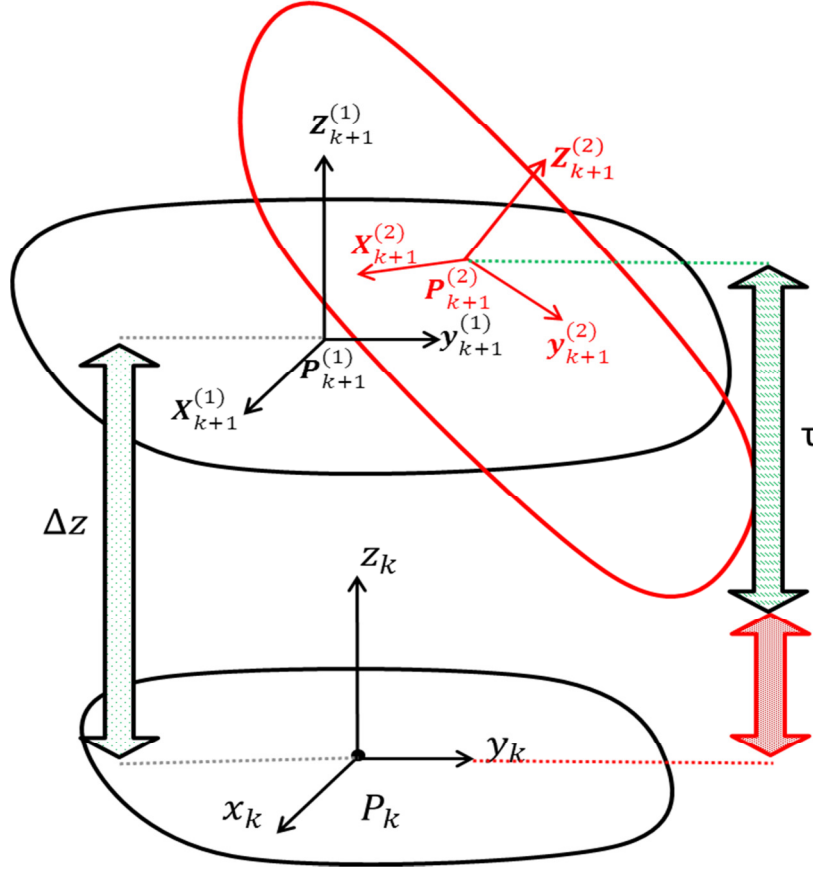


Figure 5.10: Orientation control schematic.

In the event that the computed θ and β from (5.4) are not within the desired range, a new centroid is arbitrarily obtained by a weighted average of the centroid of the previous cross-section moved straight up by Δz and the current centroid (Figure 5.11). Orientation control is repeated for a preset number of iterations (6). If the rare case that

the outcome would not be successful, the user can simply hit the “redo” button to go one step back and change Δz to ease up the process.

Inputs: Index of current cross-section (k), local unit vectors (xx, yy, zz), approximate centroid of cross-section (Center), distance between two centroids of consecutive cross-sections (deltaz), precision (eps)

Output: centroid of cross-section (New_center), updated local unit vectors (xx, yy, zz)

```

New_center = zeros(3,1); Old_center = zeros(3,1);
xx(:,k) = xx(:,k-1); yy(:,k) = yy(:,k-1); zz(:,k) = zz(:,k-1);
Center(:,k) = Center(:,k-1) + zz(:,k)*deltaz;
counter = 1;
New_center(:,1) = Center(:,k);
while (norm(Center(:,k) - New_center(:,1)) > eps || counter == 1) && counter < 6
    Center(:,k) = New_center(:,1);
    Old_center(:,1) = New_center(:,1);
    new_zz2(:,1) = (Center(:,k) - Center(:,k-1))/norm(Center(:,k) - Center(:,k-1));
    theta2 = asin(dot(new_zz2(:,1),xx(:,k)));
    beta2 = -asin((dot(new_zz2(:,1),yy(:,k)))/cos(theta2));
    inter2(:,1) = cos(beta2)*zz(:,k) - sin(beta2)*yy(:,k);
    new_xx2(:,1) = (cos(theta2)*xx(:,k) - sin(theta2)*inter2(:,1))
                  /norm(cos(theta2)*xx(:,k) - sin(theta2)*inter2(:,1));
% Dividing by the norm avoids numerical errors later when determining theta2 and beta2
    new_yy2(:,1) = cross(new_zz2(:,1),new_xx2(:,1));
    [New_center(:,1),~,xn(k,:),yn(k,:),handles.zn(k,:),Radius(:,k)] =
    Ray_tracing(Center(:,k),0.6*min(Radius(:,k-1)),
    eps,circ_c,new_xx2(:,1),new_yy2(:,1),u,v,w,calX,calY,calZ); % see Figure 5.6
    Center(:,k) = New_center(:,1);
    new_zz2(:,1) = (Center(:,k) - Center(:,k-1))/norm(Center(:,k) - Center(:,k-1));
    theta2 = asin(dot(new_zz2(:,1),xx(:,k)));
    beta2 = -asin((dot(new_zz2(:,1),yy(:,k)))/cos(theta2));
    inter2(:,1) = cos(beta2)*zz(:,k) - sin(beta2)*yy(:,k);
    new_xx2(:,1) = (cos(theta2)*xx(:,k) - sin(theta2)*inter2(:,1))
                  /norm(cos(theta2)*xx(:,k) - sin(theta2)*inter2(:,1));
% Dividing by the norm avoids numerical errors later when determining theta2 and beta2
    new_yy2(:,1) = cross(new_zz2(:,1),new_xx2(:,1));
    if abs(theta2) > asin(tortlimit*deltaz/
        (0.5*max(Radius(:,k-1))+0.5*max(Radius(:,k))))
        || abs(beta2) > asin(tortlimit*deltaz/
        (0.5*max(Radius(:,k-1))+0.5*max(Radius(:,k))))
        Center(:,k) = 0.9*Old_center(:,1) + 0.1*New_center(:,1);
        new_zz2(:,1) = (Center(:,k) - Center(:,k-1))/norm(Center(:,k) - Center(:,k-1));
    end
    New_center(:,1) = Center(:,k-1) + new_zz2(:,1)*deltaz;
    Center(:,k) = Old_center(:,1);
    counter = counter + 1;
end
xx(:,k) = new_xx2(:,1);
yy(:,k) = new_yy2(:,1);
zz(:,k) = new_zz2(:,1);

```

Figure 5.11: Marching algorithm with orientation control (in MATLAB format).

5.6 Preliminary structured mesh

The process just described results in the reconstruction of the centerline of the aorta as well as the generation of a preliminary structured mesh. Figure 5.12 shows results at different times. The mesh generation from first to last cross-section takes only a few seconds.

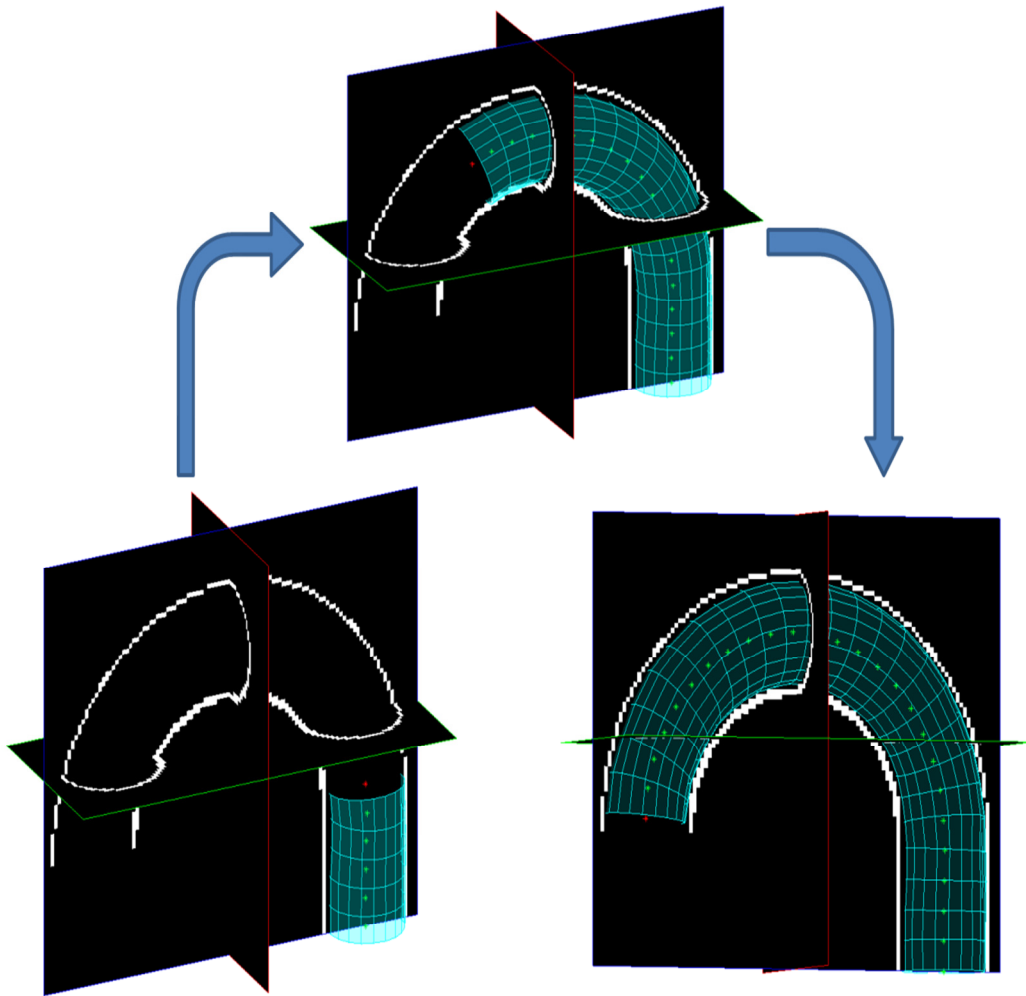


Figure 5.12: Progression of mesh generation process on synthetic aorta.

5.7 Structured mesh refinement

The preliminary centerline and mesh discussed above are meant as a first interpretation of the image volume. In simple cases such as with the synthetic aorta, the preliminary mesh may already be of good quality, but this not general. Therefore, a mesh refinement procedure was designed based on the information about the aortic geometry gathered from the preliminary steps.

One of the main issues with generating a refined structured mesh by using a centerline and variable radii turned out to be the possibility that the mesh would self-intersect or fold. The risk of intersection increases with the number of cross-sections and the tortuosity of the centerline. In other words, if few cross-sections along a tortuous centerline are present, these cross-sections are less likely to intersect. Although folding is limited by the orientation control algorithm, it may simply happen when more cross-sections along the centerline are requested by the user after the preliminary mesh is constructed.

To mitigate this potential problem, the refined mesh algorithm starts by simplifying the definition of the preliminary centerline. This is done by fitting a cubic spline through its points (say 30, corresponding to 30 cross-sections) and down-sampling them (to say, $n = 10$, corresponding to as many cross-sections). At this stage, the ray-tracing algorithm is used to find the exact centroids, coordinate systems and radii associated with these newly defined cross-sections. Each contour is discretized into as many as requested by the user in the circumferential direction, say $ncirc$. Then, $ncirc$ cubic splines are fitted through the nodes of all cross-sections, creating $ncirc$ longitudinal lines generally parallel to the centerline of the aorta. These lines are evenly cut into as

many elements as requested by the user in the longitudinal direction of the aorta. This makes sure that adjacent cross-sections do not intersect.

The value of n may have a direct impact on the quality of the final structured mesh obtained, although it does not in the case of the synthetic aorta. Therefore the GUI provides the user with the possibility of varying n until satisfactory results are obtained. The final mesh generation algorithm is presented in Figure 5.13. The mesh shown in Figure 5.14 was generated with $n = 6$.

Input: Coordinates of n_{current} points on aortic centerline (Center), user-requested number of points on centerline for downsampling (n), number of nodes requested on each cross-section (n_{circ})

Output: New nodal mesh coordinates (x_n, y_n, z_n)

```
function Mesh()
    Initilization of all variables (removed)
    % Down-sampling the centerline to n points
    XYZ = Center(:,1:ncurrent); f = cscvn(XYZ); [~,T] = fnplt(f); tmax = max(T);
    t = linspace(0,tmax,n); local.NewCenter = fnval(f,t);
    NewCenter = local.NewCenter;
    % Local coordinate systems are determined for each center point; contours are found for
    each new center point using ray-tracing method.
    V12(:,1) = NewCenter(:,2)-NewCenter(:,1);Newzz(:,1) = V12(:,1)/norm(V12(:,1));
    V13(:,1) = [1*handles.calX;1*handles.calY;1*handles.calZ] - NewCenter(:,1);
    Newxx(:,1) = cross(V13(:,1),V12(:,1))/(norm(cross(V13(:,1),V12(:,1))));
    Newyy(:,1) = cross(Newzz(:,1),Newxx(:,1));
    [~,~,Newxn(1,:),Newyn(1,:),Newzn(1,:),~] =
Ray_tracing(NewCenter(:,1),initradius,eps,ncirc,Newxx(:,1),Newyy(:,1),u,v,w,calX,calY,
calZ); % see Figure 5.6
    for k = 2:n
        Newzz(:,k) = (NewCenter(:,k)-NewCenter(:,k-1))/norm(NewCenter(:,k)-NewCenter(:,k-1));
        theta1 = asin(dot(Newzz(:,k),Newxx(:,k-1)));
        beta1 = -asin((dot(Newzz(:,k),Newyy(:,k-1)))/cos(theta1));
        inter1(:,1) = cos(beta1)*Newzz(:,k-1) - sin(beta1)*Newyy(:,k-1);
        Newxx(:,k) = cos(theta1)*Newxx(:,k-1) - sin(theta1)*inter1(:,1);
        Newyy(:,k) = cross(Newzz(:,k),Newxx(:,k));
        [~,~,Newxn(k,:),Newyn(k,:),Newzn(k,:),~] =
Ray_tracing(NewCenter(:,k),initradius,eps,ncirc,Newxx(:,k),Newyy(:,k),u,v,w,calX,calY,
calZ); % see Figure 5.6
    end
    % Cubic splines are fitted through the "i-th" node of each reconstructed cross-section,
    and npoints evenly-spaced new cross-sections are determined.
    XYZ = zeros(3,n);
    for i = 1:ncirc
        for k = 1:n
            XYZ(:,k) = [Newxn(k,i); Newyn(k,i); Newzn(k,i)];
        end
    end
    f = cscvn(XYZ); [~,T]=fnplt(f); tmax=max(T); t=linspace(0,tmax,n); newXYZ=fnval(f,t);
    for k = 1:npoints
        xxn(k,i) = newXYZ(1,k); yyn(k,i) = newXYZ(2,k); zzn(k,i) = newXYZ(3,k);
    end
end
% The centroids of the new cross-sections are determined.
for k = 1:n
    Center(1,k) = sum(xxn(k,:),2)/ncirc;
    Center(2,k) = sum(yyn(k,:),2)/ncirc;
    Center(3,k) = sum(zzn(k,:),2)/ncirc;
end
    Determine local coordinate systems determined for each center point; find
    contours for each new center point using ray-tracing method (see Figure 5.6), as above
    (removed).
end
end
```

Figure 5.13: Abridged mesh refinement algorithm (in MATLAB format).

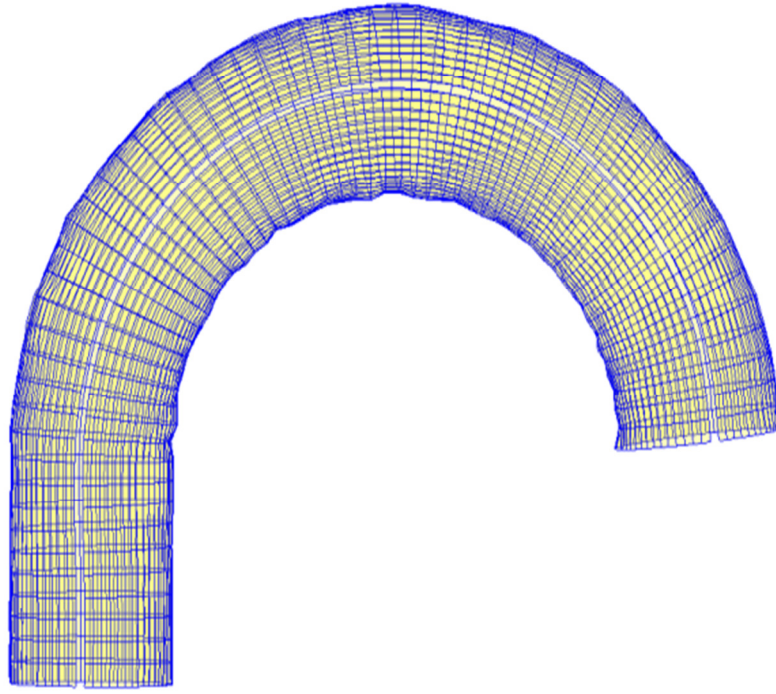


Figure 5.14: Final mesh for the synthetic aorta.

The proposed automatic surface mesh generation methodology was tested on three different datasets in addition to the synthetic aorta. This is discussed next.

5.8 Test on the Visible Human dataset

The Visible Human dataset was made available to us through Visible Human Project license agreement (U.S. National Library of Medicine, 2013). In brief, the body of a male subject was frozen and cut axially into at 1 mm intervals. Photographs were taken of each of the 1,871 cross-sections for whole body. The photographs were digitized at a full scale resolution of 4096 pixels by 2700 pixels, and formatted as .jpg files. However, the images are not well oriented. Therefore, we first had to realign the whole stack of images to

make them useable. This was done by writing a MATLAB script to give the user options to align individual images by moving them up, down, right and left until visual alignment was satisfactory (Figure 5.15).

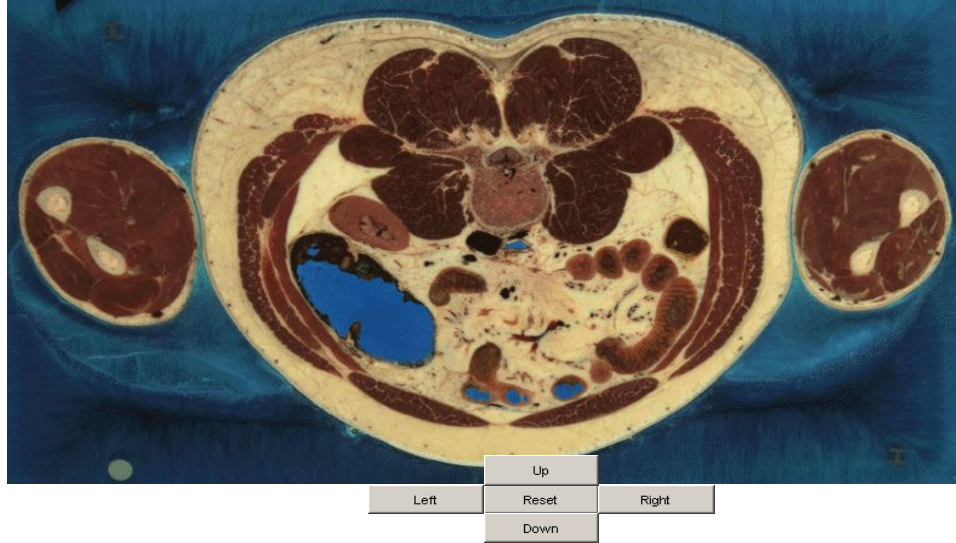


Figure 5.15: An original image from the Visible Human dataset (U.S. National Library of Medicine, 2013).

Once proper alignment was reached, and after blue artefacts present in many images were manually removed, the whole dataset was formed by stacking up, calibrating and saving the images for later usage. Table 5.1 shows the calibration values used to match image pixels to MATLAB voxel domain.

Table 5.1 Voxel dimensions for Visible Human dataset.

Direction	Calibration
X	.56 mm
Y	.56 mm
Z	1 mm

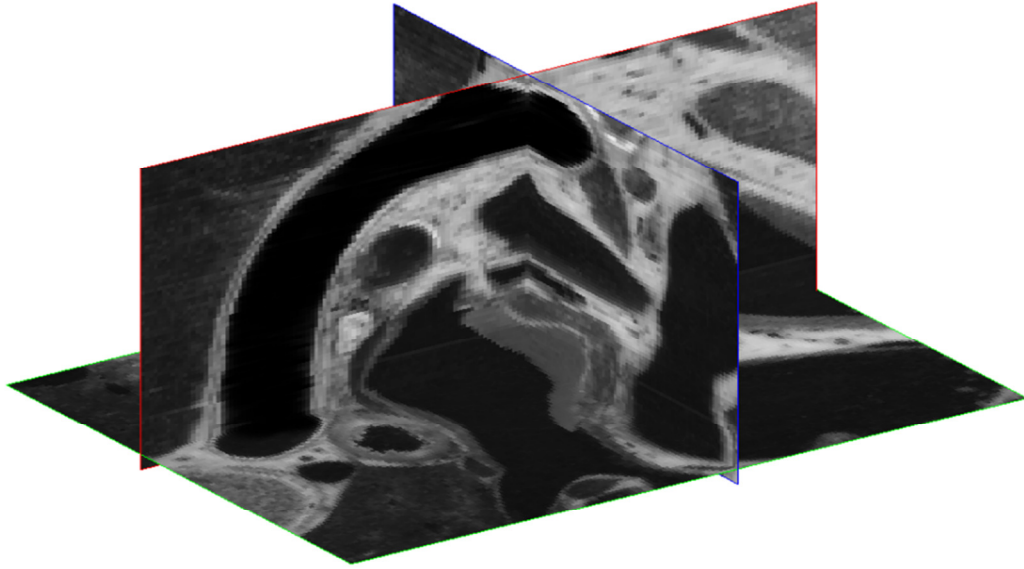


Figure 5.16: 3-D view of the cropped volume for the Visible Human dataset.

Figure 5.16 demonstrates a 3-D view of the cropped volume. Eighty reoriented images were used to generate this volume.

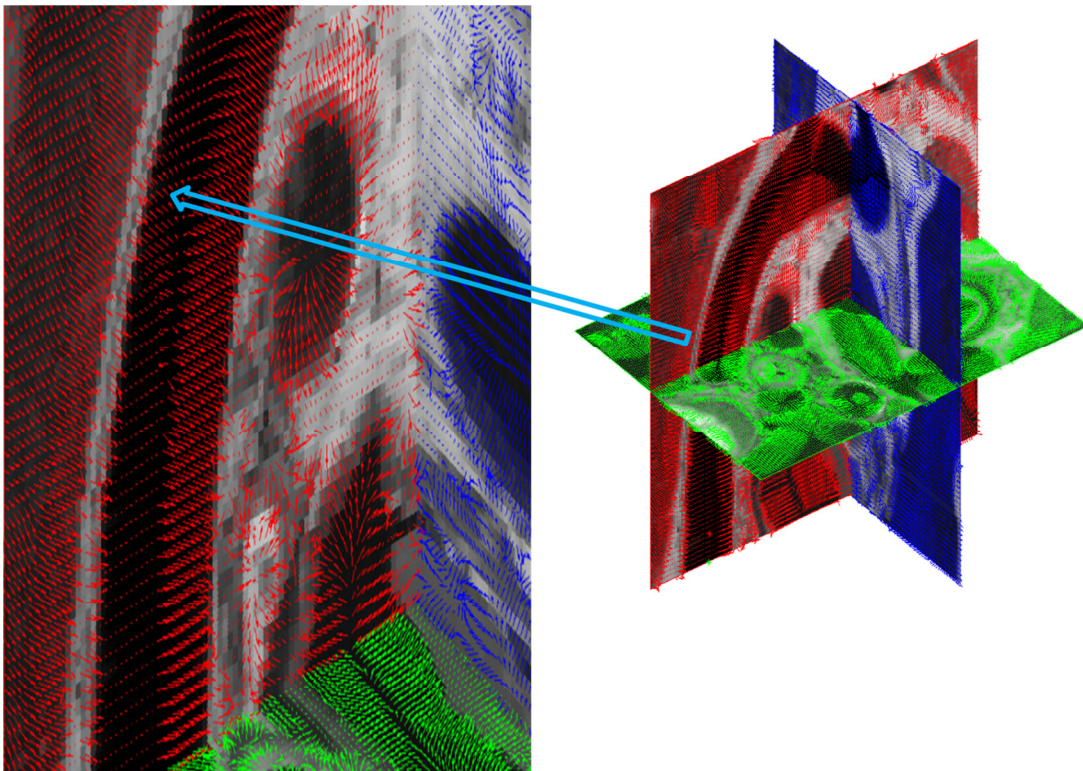


Figure 5.17: 3-D GVF field after 20 iterations, with $\mu = 0.1$.

Figure 5.17 shows the 3-D GVF field in the axial, sagittal and coronal planes as obtained after 20 iterations. Computation of the GVF took approximately 2 minutes. The GVF field near the aortic wall is magnified in the left image to make the directions of the vectors pointing to the aortic walls more visible.

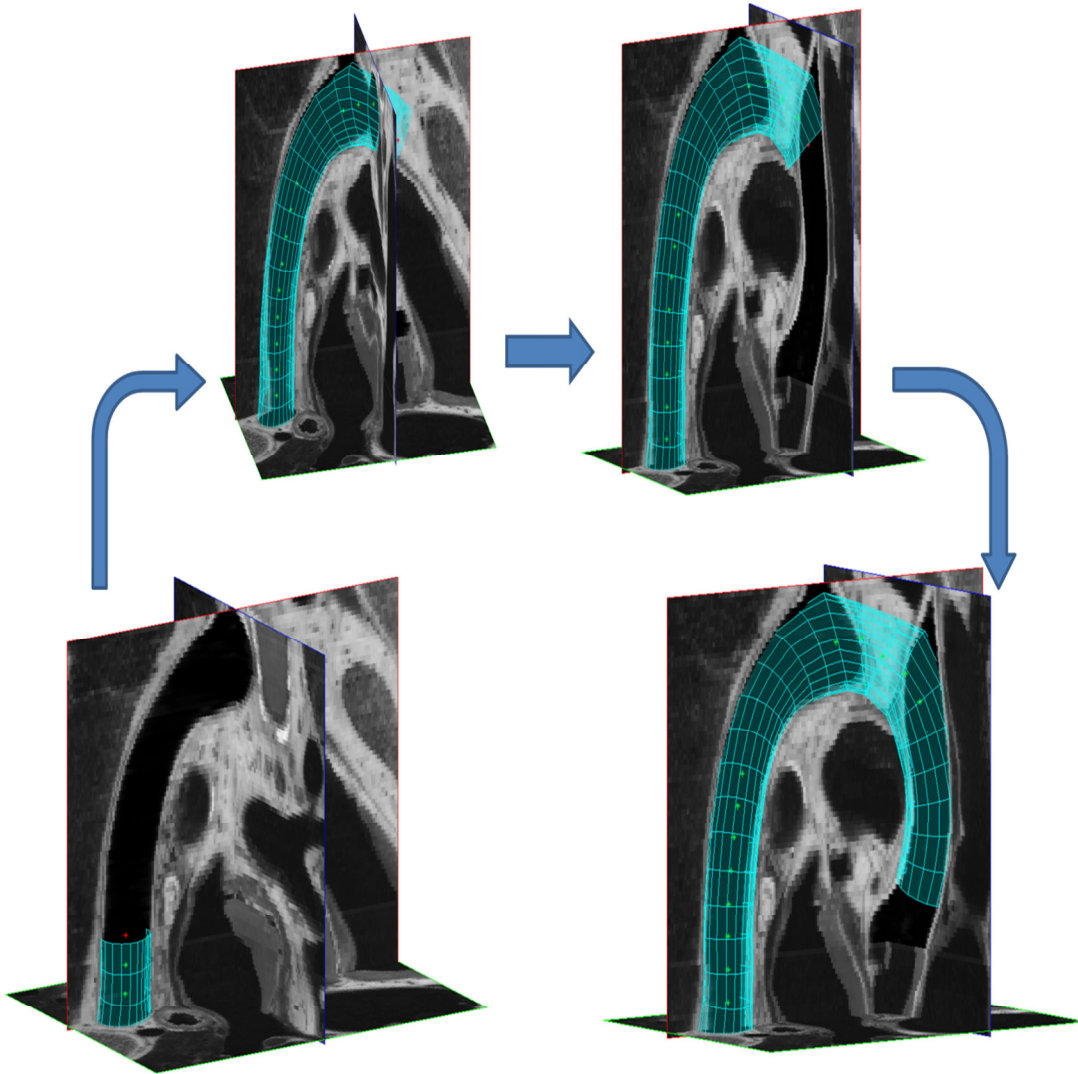


Figure 5.18: Steps of the preliminary mesh generation for the Human Visible dataset.

Different stages of the preliminary mesh generation after input of a starting point and a direction are shown in Figure 5.18 from bottom left to right. As the user pushes the “continue” button, the surface mesh is generated in cross-sectional planes. The initial mesh radius was 6 mm and tortuosity limit was .65. The processing time at this stage is on the order of 2-4 seconds for each cross-section. Overall, for a distance of 6 mm between centroids of cross-sections, the Human Visible dataset took 1-1:30 minutes for 25-30 cross-sections. This time does not include the 3-D GVF computation.

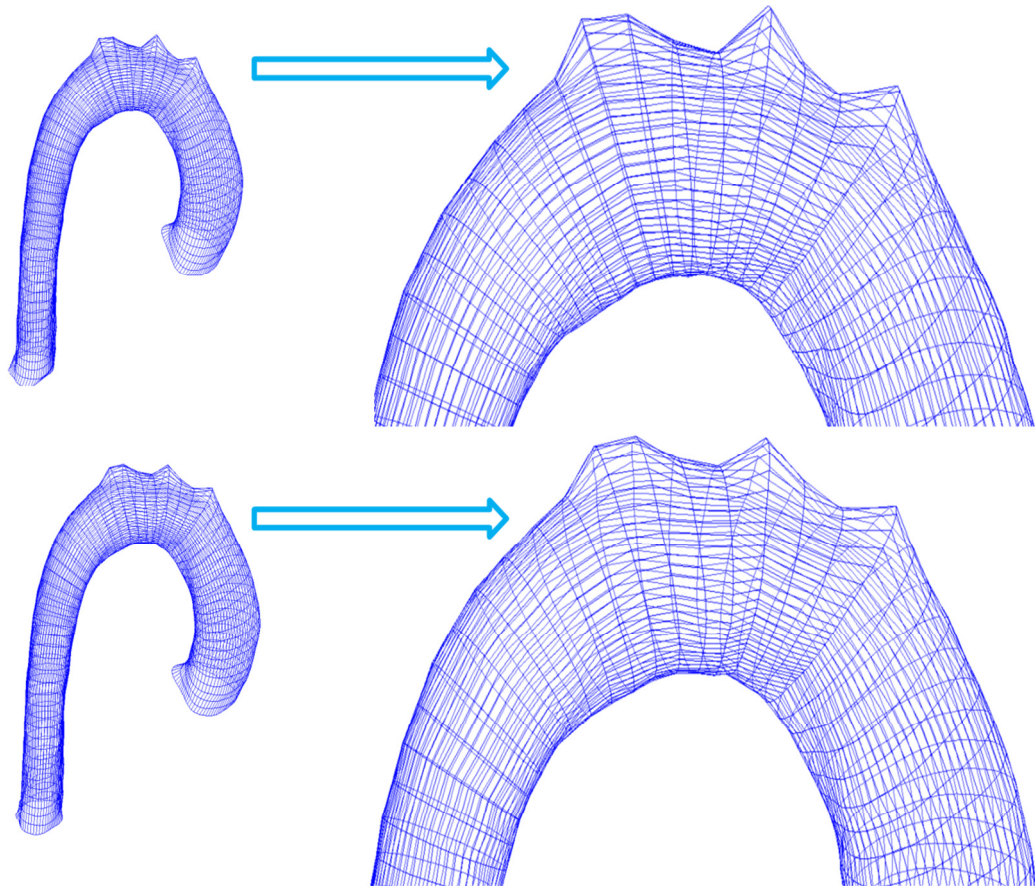


Figure 5.19: The effect of varying n ; top: $n=15$, bottom: $n=6$.

Down-sampling the number of points on the centerline was successful in removing folds in the mesh. However, the actual value for n could vary between 15 and 6 with no significant effect on the mesh, as illustrated in Figure 5.19.

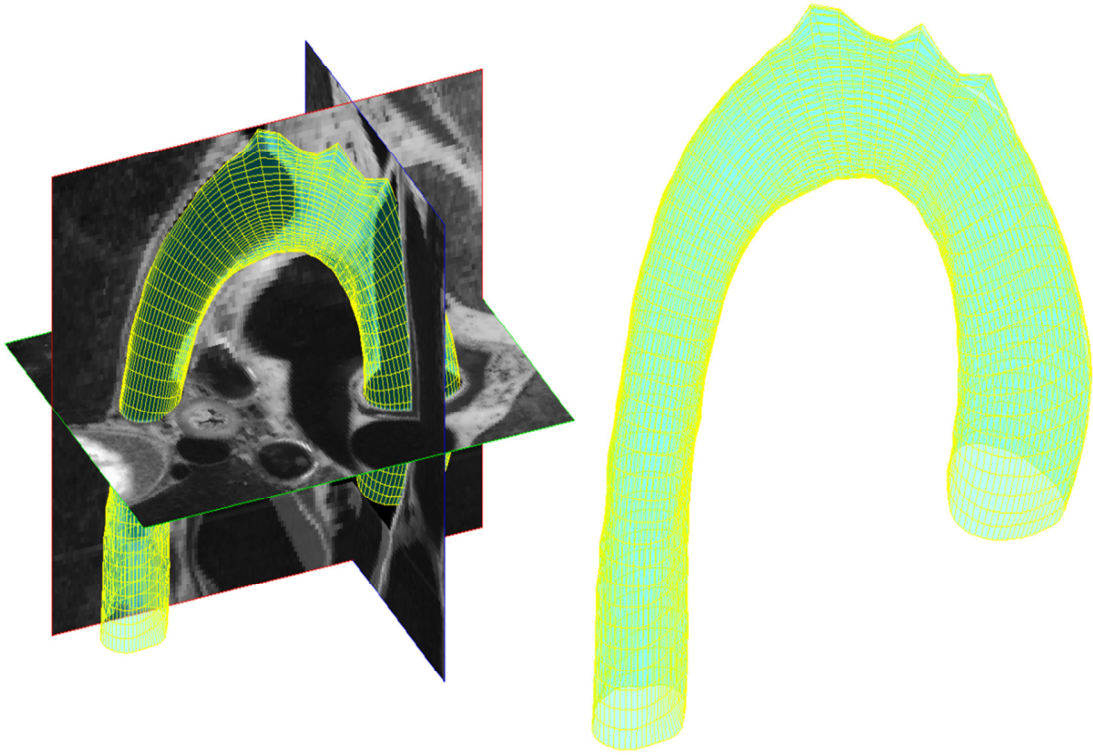


Figure 5.20: Final surface mesh for the Human Visible dataset.

The final inner surface mesh is shown in Figure 5.20 where there are 60 nodes in each cross-section's circumference in 60 cross-sections resulting in 3600 nodes in total and 3540 quad elements. The number of nodes could be easily changed. Figure 5.21 shows two other meshes, one with 100 nodes in the circumference and 60 cross-sections, and the other with 50 nodes in the circumference and 70 cross-sections.

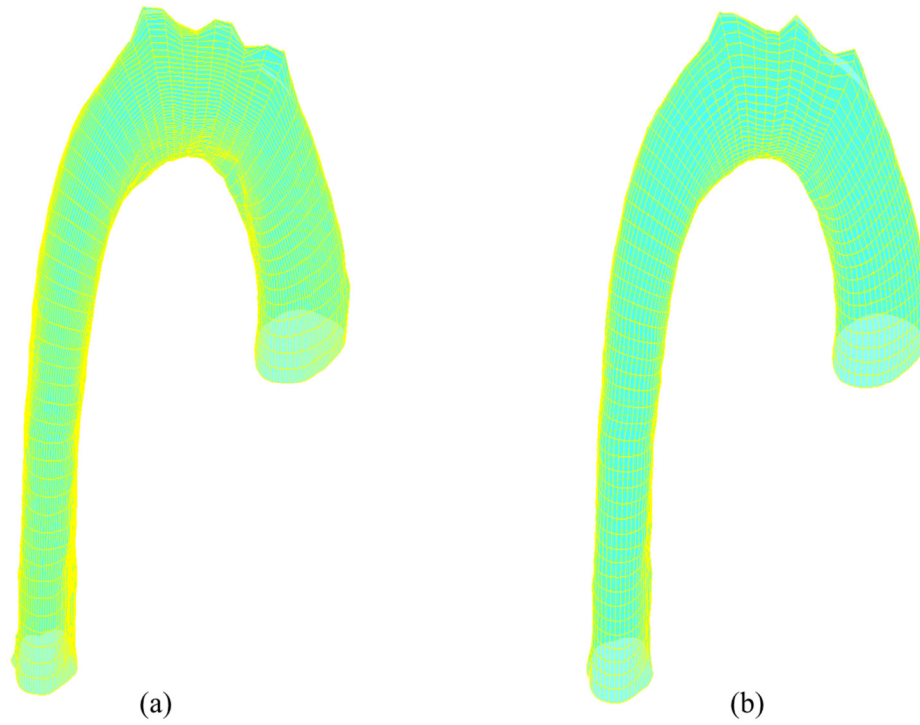


Figure 5.21: Final surface mesh for the Human Visible dataset. (a): 100 nodes in circumference and 60 cross-sections, (b): 50 nodes in circumference and 70 cross- sections.

5.9 Test on CT dataset with descending aortic aneurysm

In this section, the proposed methodology is tested on CT images of a descending aortic aneurysm stored in the DICOM format. The de-identified data were obtained from Dr. C. Beller from the University of Heidelberg, Germany. The voxel size of the dataset was 0.3 by 0.3 by 0.3 mm.

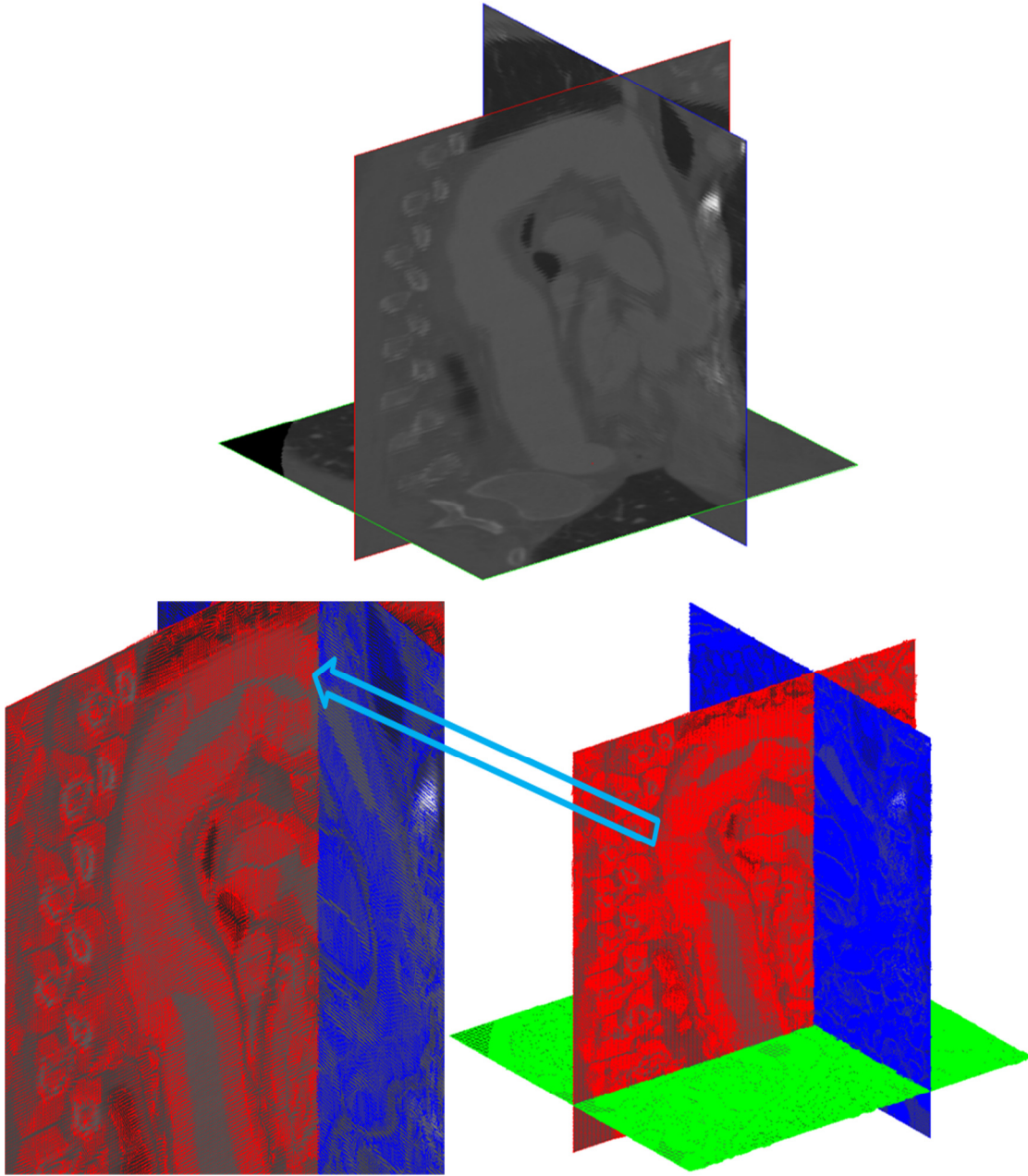


Figure 5.22: top: 3-D view of ROI, bottom: 3-D GVF field after 20 iterations, with $\mu=0.1$.

The ROI is shown in the top section of Figure 5.22, and the corresponding 3-D GVF field is represented in the bottom section. The computation time for the 3-D GVF, including the initial Canny-edge filtering, was about 4 minutes for 20 iterations.

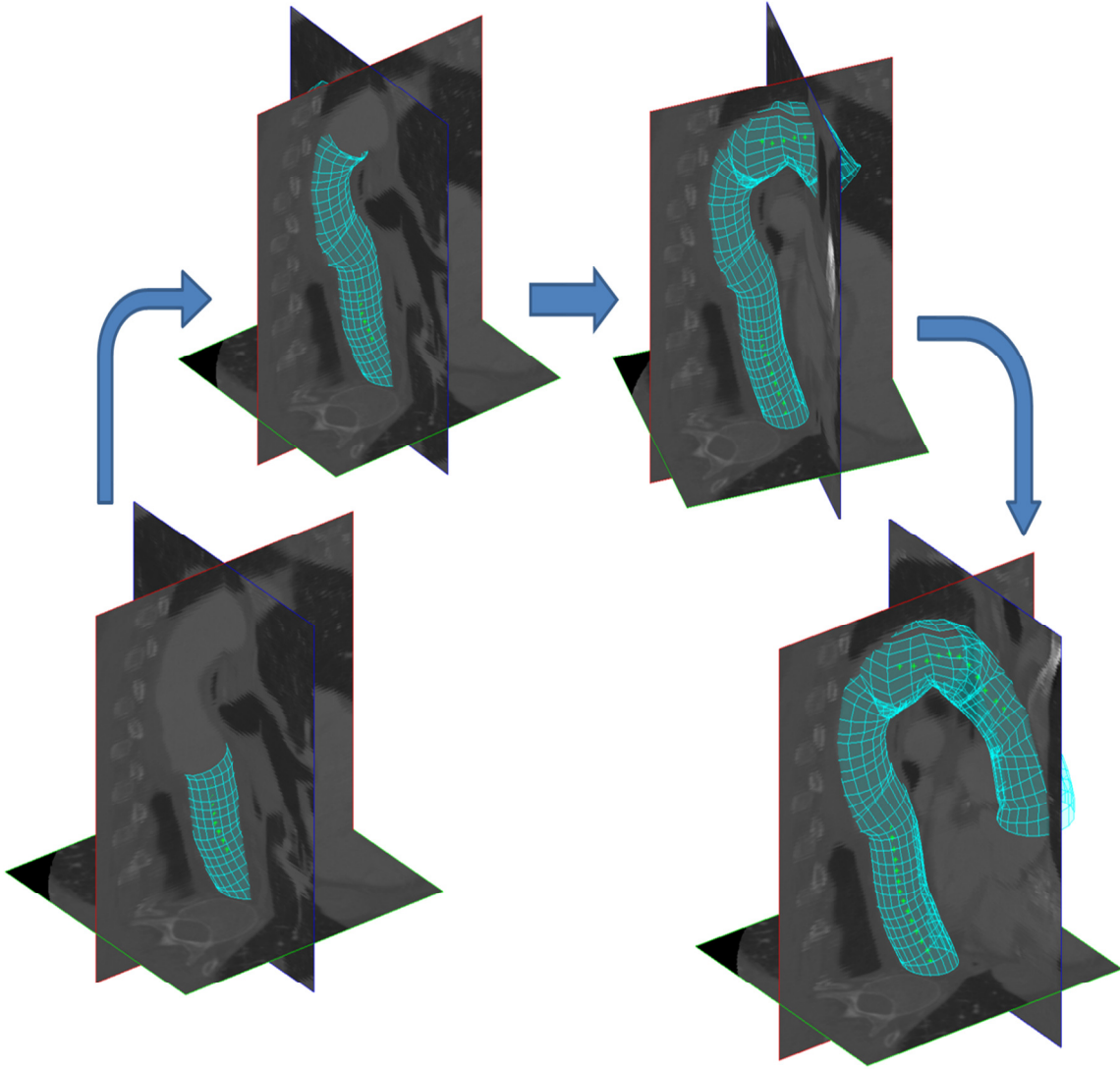


Figure 5.23: Steps of the preliminary mesh generation for dataset with descending aortic aneurysm.

Different stages of the preliminary mesh generation are shown in Figure 5.23. The initial mesh radius was 6 mm and the tortuosity limit 0.85. The processing time at this stage was on the order of 2-5 seconds for each reconstructed cross-section. The total preliminary mesh generation time was approximately 2 minutes for 55 cross-sections. As can be seen from Figure 5.26, orientation control was only partially successful, as some

adjacent cross-sections of the preliminary mesh do intersect due to high local tortuosity of the aortic centerline.

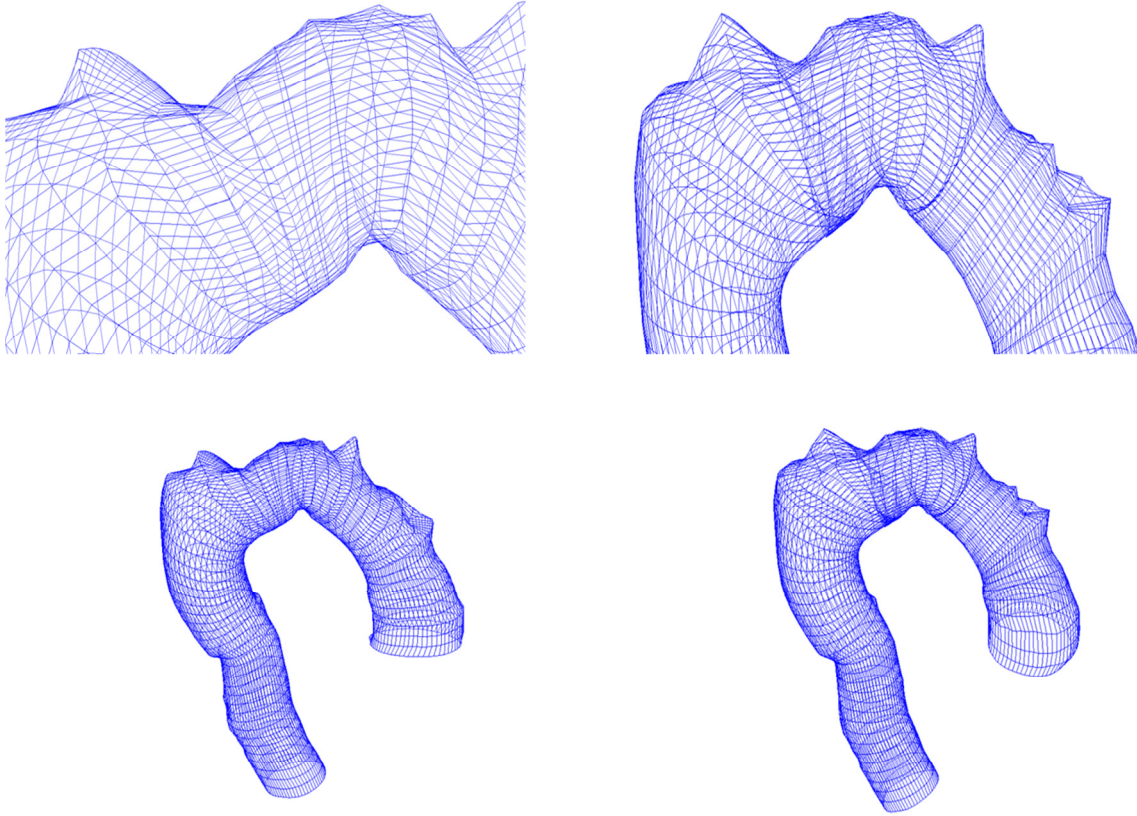


Figure 5.24: The effect of varying n ; left: $n = 6$ (no folding); right: $n = 15$ (folding).

This issue was resolved in the final mesh by down-sampling the number of points on the centerline. Figure 5.24 shows that $n = 15$ was not satisfactory, but $n = 6$ was: the mesh presents smooth edges and no folding.

The final surface mesh is shown in Figure 5.25 where there are 60 nodes in each cross-section's circumference and 60 cross-sections resulting in 3600 nodes in total and 3540 quad elements. The results for 80 nodes in the circumference and 60 or 80 cross-sections are shown in Figure 5.26. As illustrated, the methodology can adequately

segment the lumen of the aneurysm and reconstruct its geometry even with significant radial variations with a cross-section or between adjacent cross-sections.

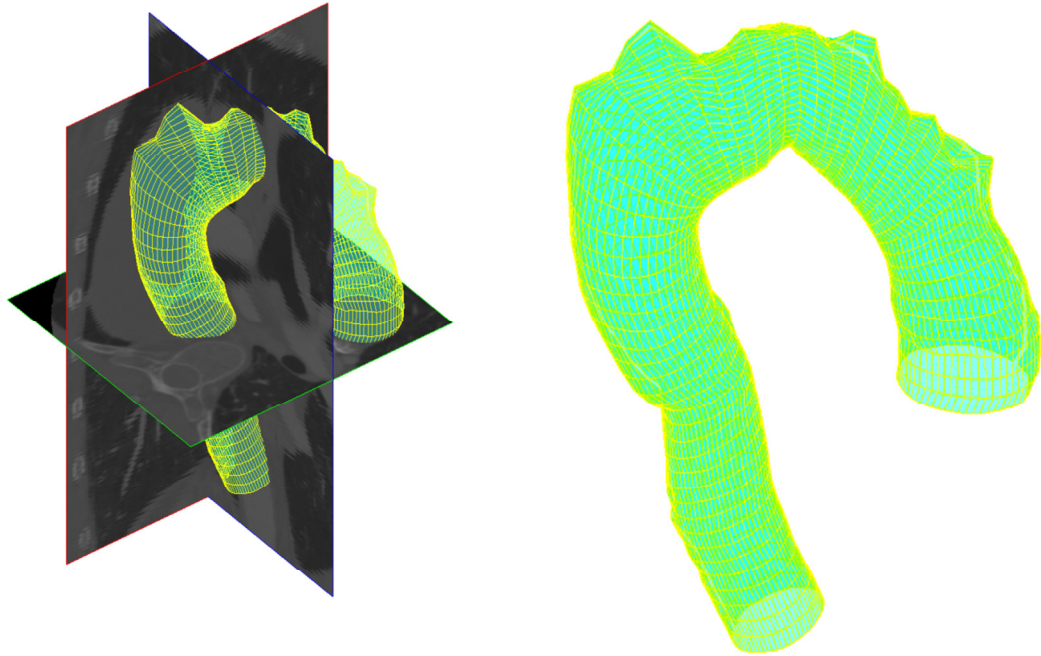


Figure 5.25: Final surface mesh with 60 nodes in circumference and 60 cross-sections.

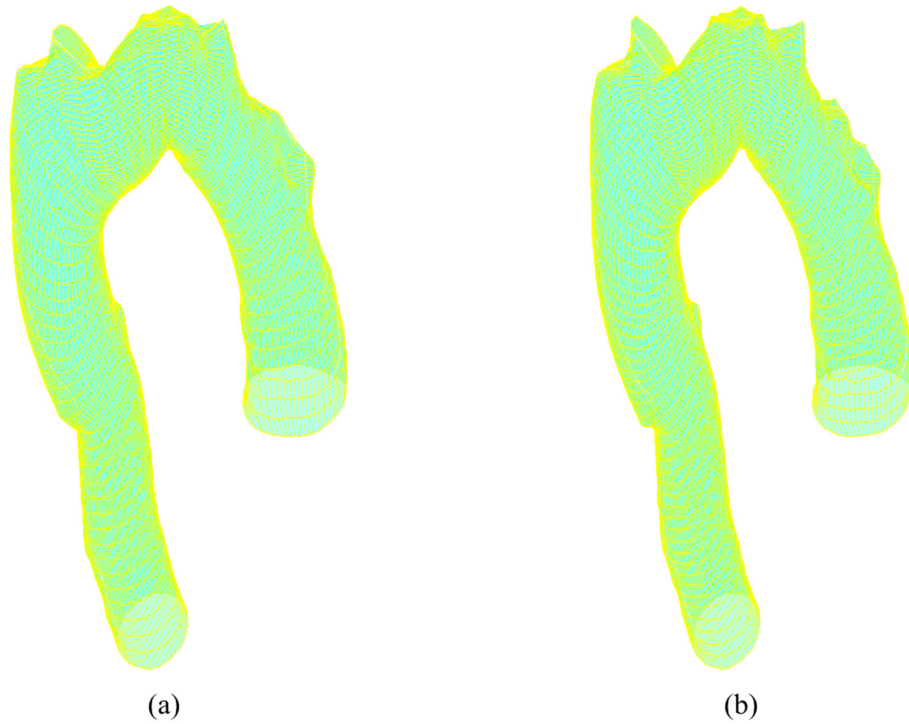


Figure 5.26: Final surface mesh: (a): 80 nodes in circumference and 60 cross-sections, (b): 80 nodes in circumference and 80 cross-sections.

5.10 Test on CT dataset with thoracic aortic aneurysm

The following dataset is the “Artifix” dataset made available for research and teaching purposes by (OsiriX Team, 2012) as DICOM samples. The dataset consists in de-identified CT images of a dilated thoracic aorta. The voxel dimensions are presented in Table 5.2.

Table 5.2 Voxel dimensions for thoracic aortic aneurysm dataset.

Direction	Calibration Value
X	.586 mm
Y	.586 mm
Z	1.5 mm

The ROI and the corresponding 3-D GVF field are shown in Figure 5.27. This dataset had less noise compared to the previous one. Consequently, the 3-D GVF quality was higher, which in turn produced better mesh results. The 3-D GVF field computation time was less than 4 minutes for 20 iterations.

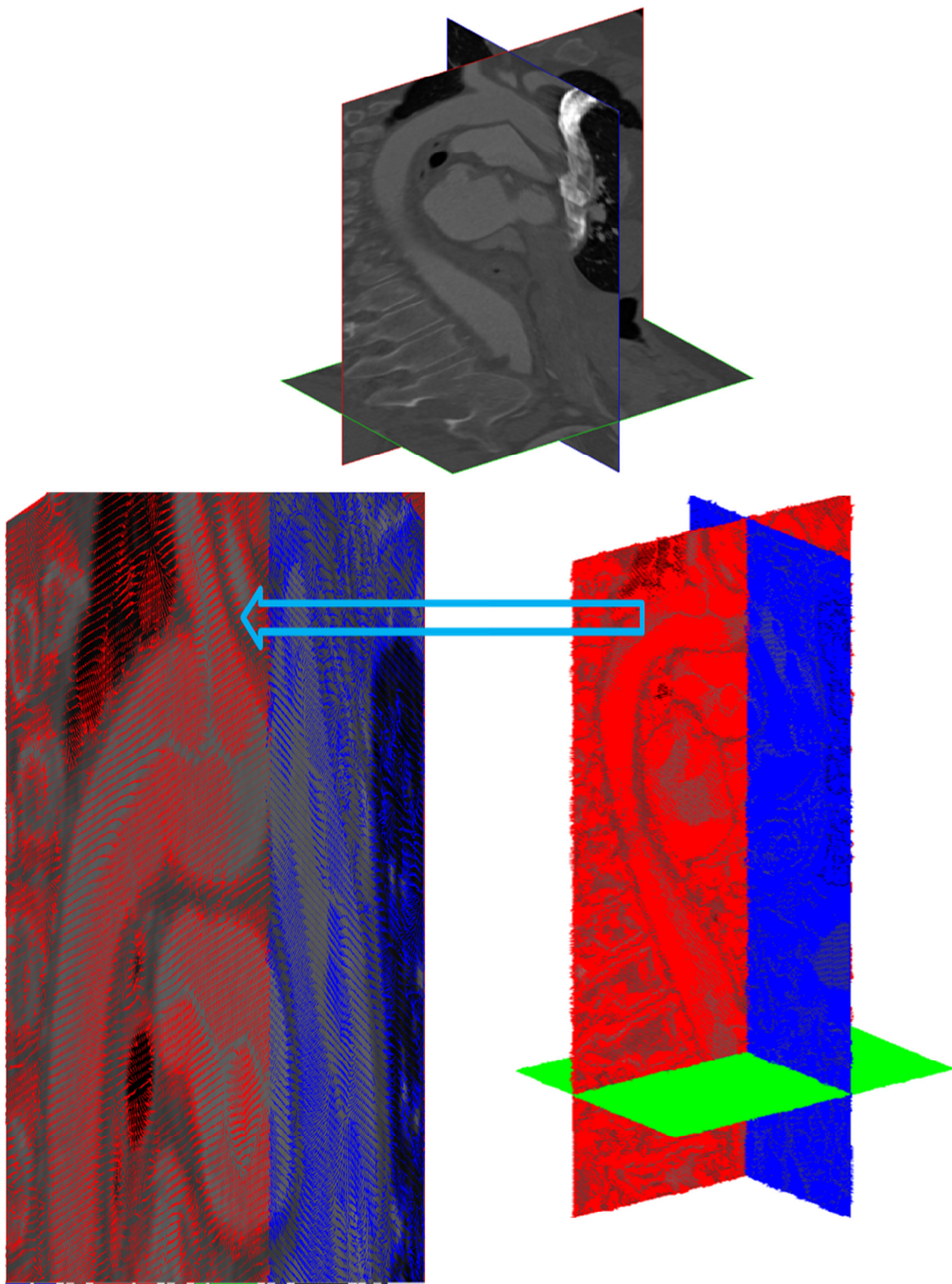


Figure 5.27: top: 3-D view of the ROI, bottom: 3-D GVF field after 20 iterations, with $\mu=0.1$.

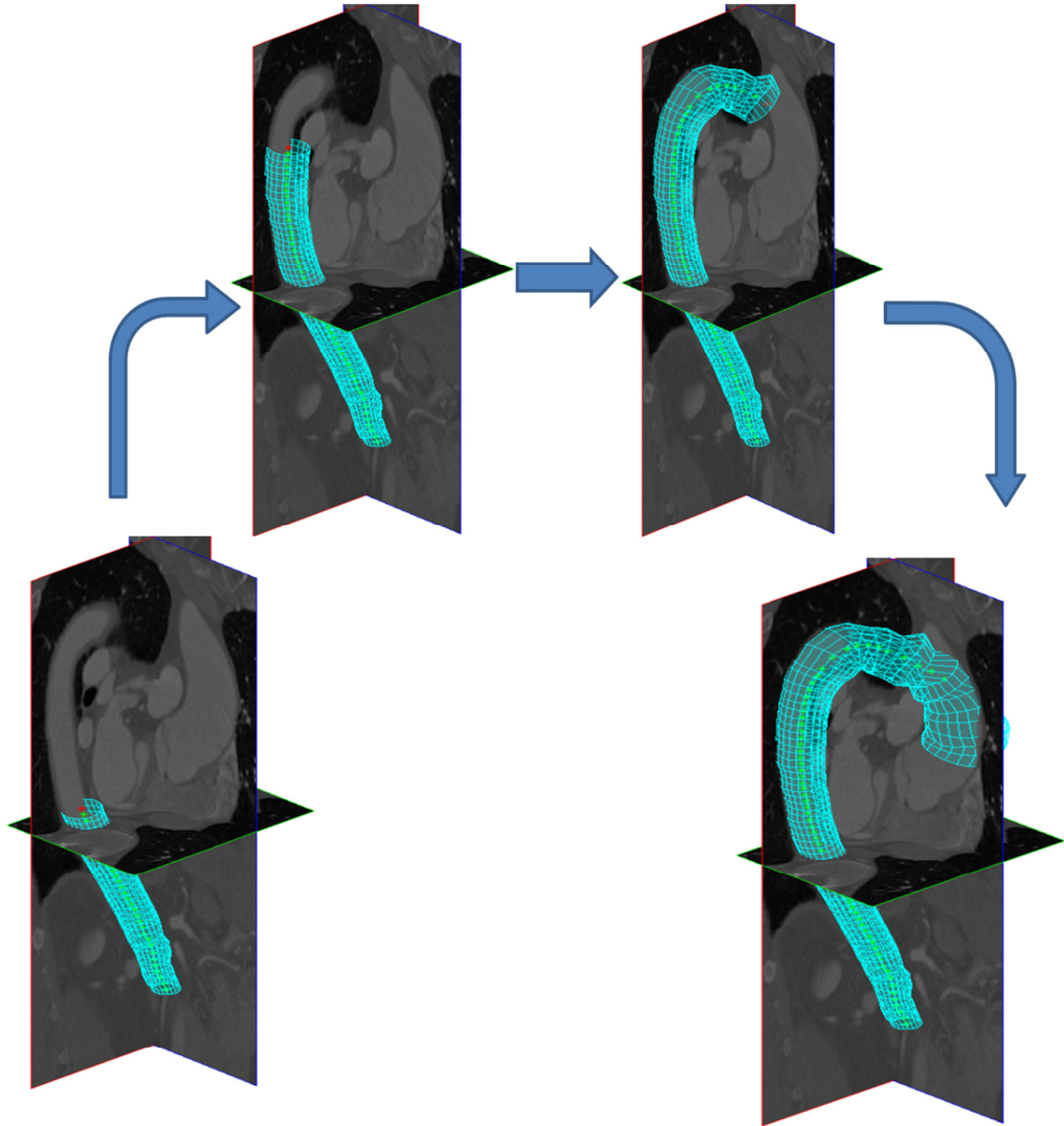


Figure 5.28: Steps of the preliminary mesh generation for dataset with thoracic aortic aneurism.

The preliminary mesh generation process is shown in Figure 5.28. The initial mesh radius was 6 mm and the tortuosity limit 0.65. The distance between centroids of cross-sections (Δz) was initially set at 8 mm in the abdominal aorta. It was increased to 12 mm in the aortic arch area for easier progress, and decreased again to 8 mm in the ascending aorta. Again, computations in one cross-section took on the order of seconds.

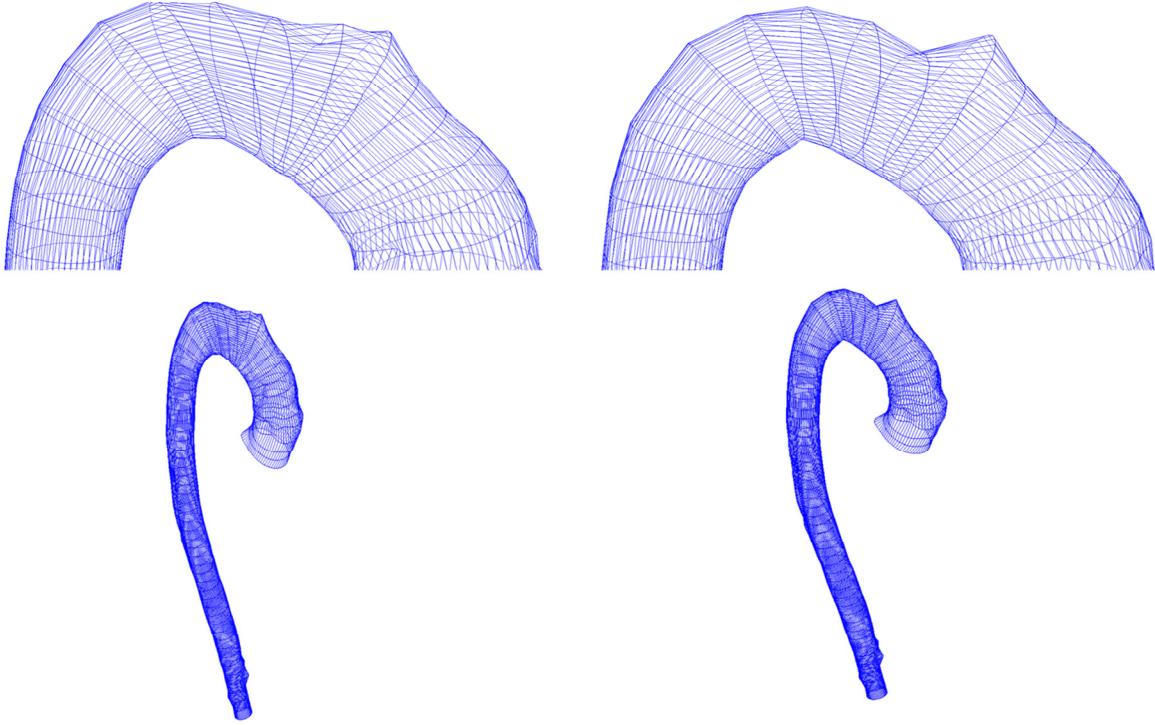


Figure 5.29: The effect of varying n ; left: $n = 10$, right: $n = 11$.

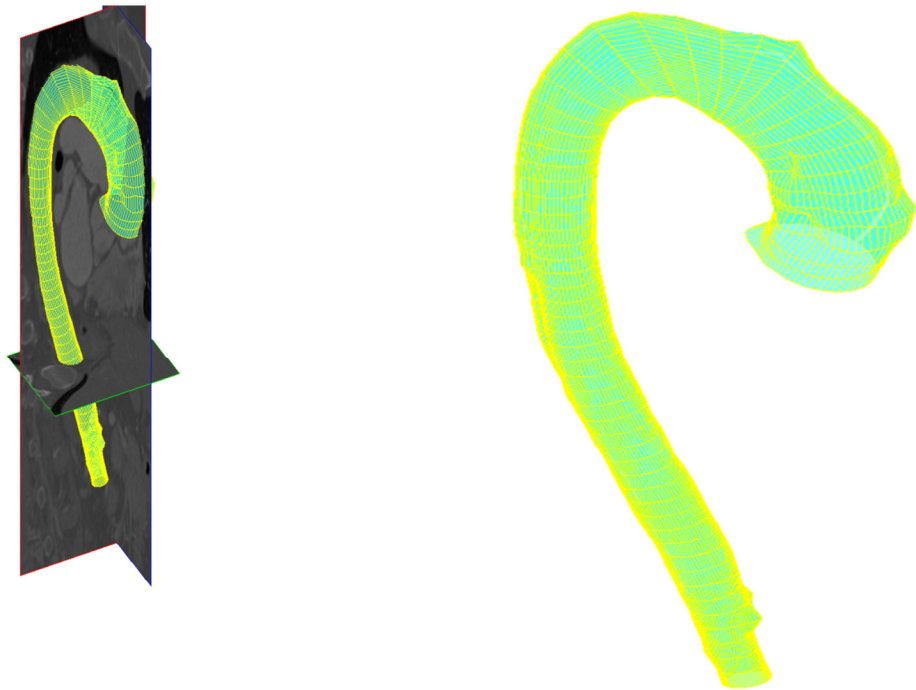


Figure 5.30: Final surface mesh with 60 nodes in circumference and 60 cross-sections.

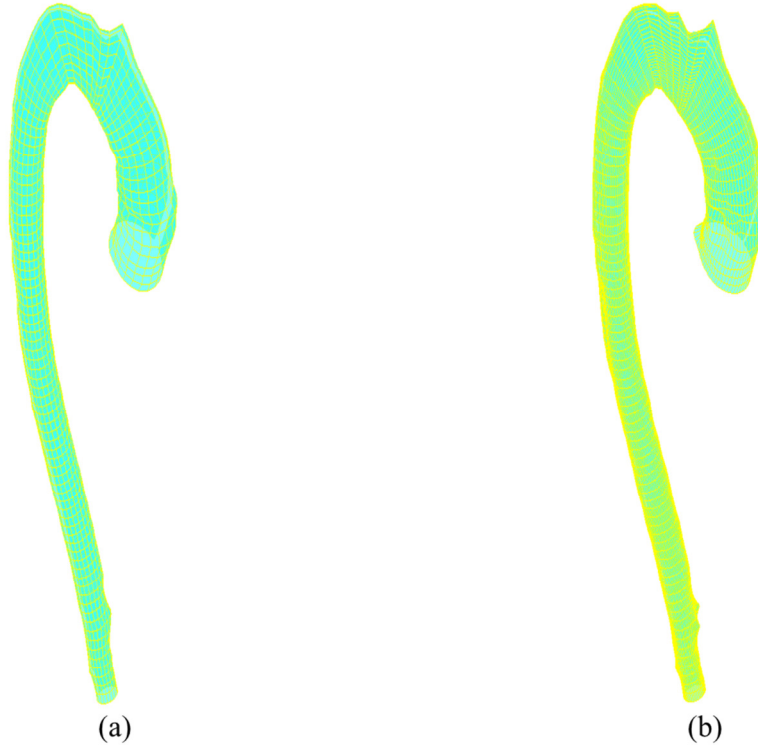


Figure 5.31: Final surface mesh: (a) 20 nodes in circumference and 100 cross-sections, (b) 60 nodes in circumference and 100 cross-sections.

Two of the best results obtained by varying the level of down-sampling of the number of points on the centerline generated are shown in Figure 5.29. In both cases, there was no folding or intersections in the reconstructed aorta. The resulting final surface mesh with 60 nodes in the circumference and 60 cross-sections is shown in Figure 5.30. The different mesh sizes are shown in Figure 5.31.

5.11 Discussion of results

As illustrated in the examples above, the proposed methodology yields structured surface meshes of the aorta in typically in less than 5 minutes, which is extremely fast compared to other existing methods. This time includes the 3-D GVF computation which is done

with little to no input from the user as the default value of the regularization parameter can be set at 0.1. The user mostly has to decide where to start the reconstruction and where to stop it, and may only have to alter the default value of the distance between two consecutive cross-sections and/or the default value of the parameter controlling the tortuosity of the centerline of the reconstructed aorta. This is in sharp contrast with other existing methods where multiple parameters may need adjustments.

One should also notice that, similar to the other existing segmentation methods, the new methodology can adequately determine the contour of the aortic lumen but, in its current version, may not be able to find the aortic wall when it is covered by thrombus. This is visible in Figure 5.23 of the descending aortic aneurysm with intra-luminal thrombus. The issue has been addressed in the literature by carrying out two segmentations, one for the lumen, and one for the outermost layer of the aortic wall (Ukwatta 2013). Then, the space between the two contours can be back-filled with the aortic wall (of assumed thickness) and the thrombus. A similar strategy may be used with the present methodology at a later stage. Before then, it was deemed important to evaluate other aspects of the results provided by the methodology. One aspect discussed in the next section is the output of the mesh to a third-party FE code toward actual analysis. Other issues addressed in the next chapters are mesh quality, the ability to generate structured meshes for the aortic wall and the blood, as well as the possibility to connect several meshes to cover the major branches of the aorta.

5.12 Output to third-party FE code

The purpose of this work is to generate finite element meshes to be used for structural and/or fluid analyses. The meshes produced above are for now just wireframes that MATLAB can display. To become understandable by commercial FE software packages such as LS-DYNA, ANSYS, ABAQUS etc., the meshes need to be transformed into a list of numbered nodes and a connectivity table, wherein node numbers are assigned to numbered elements, in a specific order. The process of generating these lists is similar for any software package. In the case at hand, the surface mesh is meant to be used with quadrilateral shell FE elements. At its most basic, a quadrilateral shell element is geometrically described by the locations of its four corner nodes. The actual element type (including the number and nature of degrees of freedom assigned to each node) is beyond the scope of the present work.

Numbering of the nodes in the final mesh is done automatically at the end of the image processing: the nodes on the first cross-section are numbered from 1 to $ncirc$, those on the second cross-section are numbered from $ncirc + 1$ to $2ncirc$, and so on. What remains to be created is the connectivity table. The connectivity table generation for four-noded shell elements is described in Figure 5.32 and 5.33.

The node locations and the connectivity table can be easily exported to third-party FE code. As an illustration of the process, the meshes generated from the datasets discussed in the previous sections were exported to commercial FE software package LS-DYNA (LSTC, Las Positas, CA, USA), as shown in Figures 5.38 to 5.40. Material properties, load and boundary conditions should be added for structural or fluid analysis.


```

Inputs: Number of nodes along circumference ( $ncirc$ )
           Number of cross-sections ( $n$ )

Outputs: Number of elements ( $nelem$ )
            Connectivity table ( $E(i), N1(i), N2(i), N3(i), N4(i)$ )

Begin
 $i = 1$ 
/* Generate connectivity tables for nodes along circumference
   FOR  $k = 1: ncirc-1$ 
       FOR  $j = 1:n-1$ 
           Assign  $N1(i), N2(i), N3(i), N4(i)$ 
            $E(i) = i$ 
            $i = i + 1$ 
       ENDFOR
   ENDFOR
/* Generate connectivity table for nodes on different cross sections
   FOR  $m = 1: (n-1)$ 
       Assign  $N1(i), N2(i), N3(i), N4(i)$ 
        $E(i) = i$ 
        $i = i + 1$ 
   ENDFOR
 $nelem = i - 1$ 
End

```

Figure 5.32: Quadrilateral element connectivity table creation algorithm.

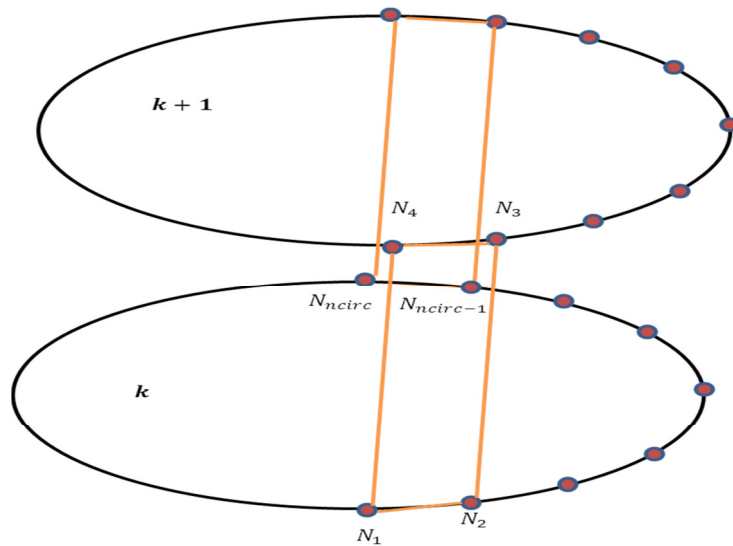


Figure 5.33: Quadrilateral element node allocation process.

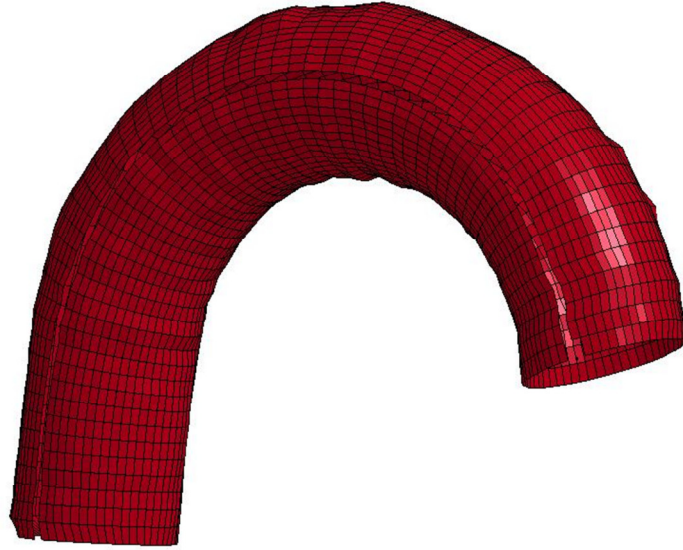


Figure 5.34: LS-DYNA mesh for synthetic aorta.

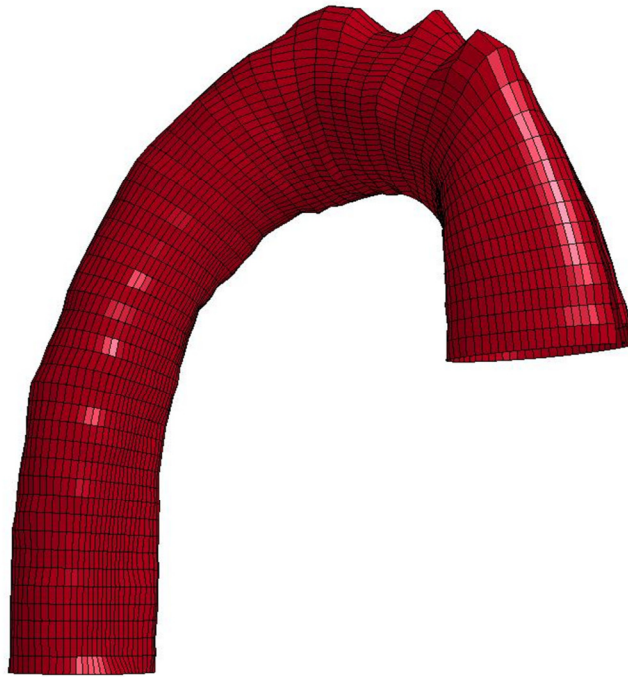


Figure 5.35: LS-DYNA mesh for the Visible Human dataset.

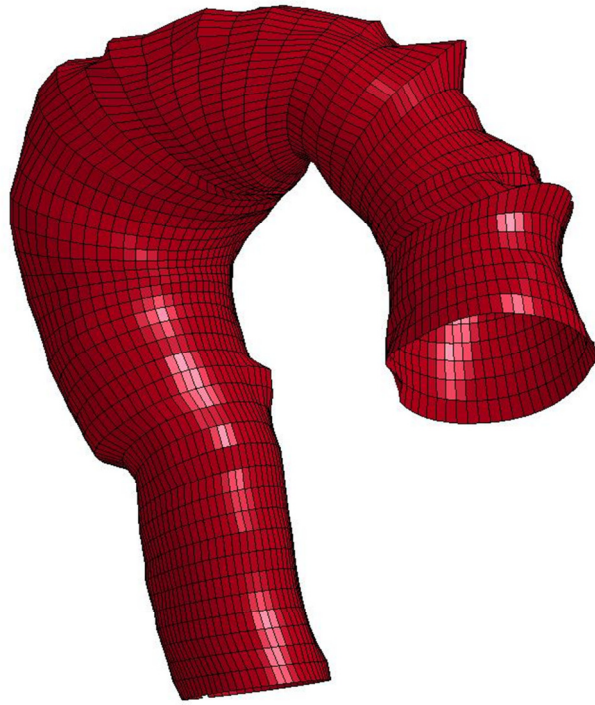


Figure 5.36: LS-DYNA mesh for descending thoracic aortic aneurysm dataset.



Figure 5.37: LS-DYNA mesh for thoracic aortic aneurysm dataset.

6 Evaluation of surface mesh quality

Before shifting our attention to volume mesh creation or meshing of the aortic branches, it is important to thoroughly assess the quality of the surface mesh obtained from the methodology presented so far. This is the purpose of the present chapter. Both reconstruction accuracy and element geometry have to be carefully considered because a very good mesh from one perspective may be a poor one from the other. Yet, a new segmentation and meshing methodology needs to satisfy both criteria if it is to be credible toward detailed structural or CFD analyses of the aorta.

6.1 Reconstruction accuracy

6.1.1 Visual inspection

For the medical datasets in general and in particular those considered in Chapter 5, there is really no information available that can be used to verify the accuracy of the reconstructed mesh. When they examine medical images, radiologists, by training and experience, know how to interpret them in view of the assets and limitations of the imaging modality. For instance, CT imaging with contrast agent is known to give more detailed information about the aortic lumen contour than MRI, because the contrast-enhanced blood shows clearly. However, neither method may be very good at showing the aortic wall thickness and/or differentiating possible intraluminal thrombus from the aortic wall. However, imaging capabilities, including their spatial and temporal resolutions, keep increasing for both imaging techniques. Therefore, as a first approach, one may simply want to make sure, by simple visual inspection, that the reconstructed

mesh of the aortic lumen does match the medical images. The GUI built around the methodology presented in Chapter 5 was designed especially for this purpose, allowing the user to check at all times and in any of the three anatomical planes, how well the boundaries of the aorta and those of the reconstructed mesh match. For instance, Figure 5.18 and Figure 5.20 in Chapter 5 demonstrate the excellent match between the boundaries of the Visible Human dataset and those of the reconstructed mesh. Figures Figure 5.23 and Figure 5.28 demonstrate the same in two CT datasets. Similarly, Figure 6.1 visually illustrates how well the reconstructed mesh fits the internal side of the aortic wall.

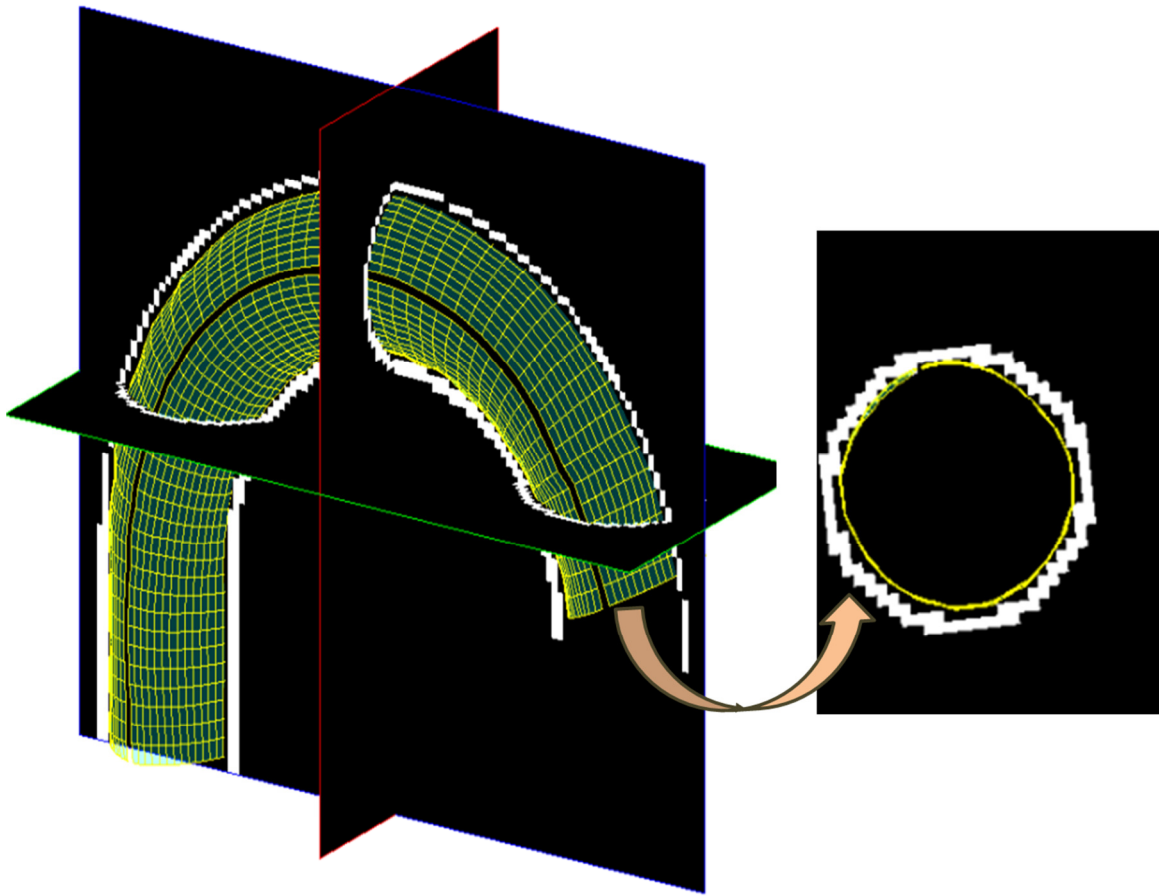


Figure 6.1: Mesh accuracy inspection: Boundary comparison for the synthetic aorta.

6.1.2 Phantom creation

Although visual inspection provides valuable qualitative feedback concerning the reconstruction accuracy, researchers have devised quantitative methods to measure it. These methods generally rely on the manufacturing of an object of known dimensions, called phantom, that is scanned using a specific medical imaging modality. The phantom is often made to closely resemble the organ of interest, in size and shape. Then, image processing of the phantom dataset is carried out, and measurements are made of the reconstructed object for comparison with the known dimensions (Lu et al., 2009; Forsyth et al., 2011; Boutchko et al., 2013).

In the following, the manufacturing of a phantom and its imaging were side-stepped due to time constraints. Instead, we assumed we knew a true surface mesh of the inner wall of the (phantom) synthetic aorta that had been used to generate the 3-D dataset. The phantom mesh was generated using the algorithm described in Figure 4.1. The outcome was a synthetic mesh with 60 nodes in each cross-section. This number of nodes was chosen to match that of the reconstructed mesh. Figure 6.2 presents the generated phantom and reconstructed meshes. The black surface with the white circle demonstrates an axial view of the aorta that was used for reconstruction. Figure 6.3 shows both individual views of the phantom and reconstructed meshes embedded in the aortic image.

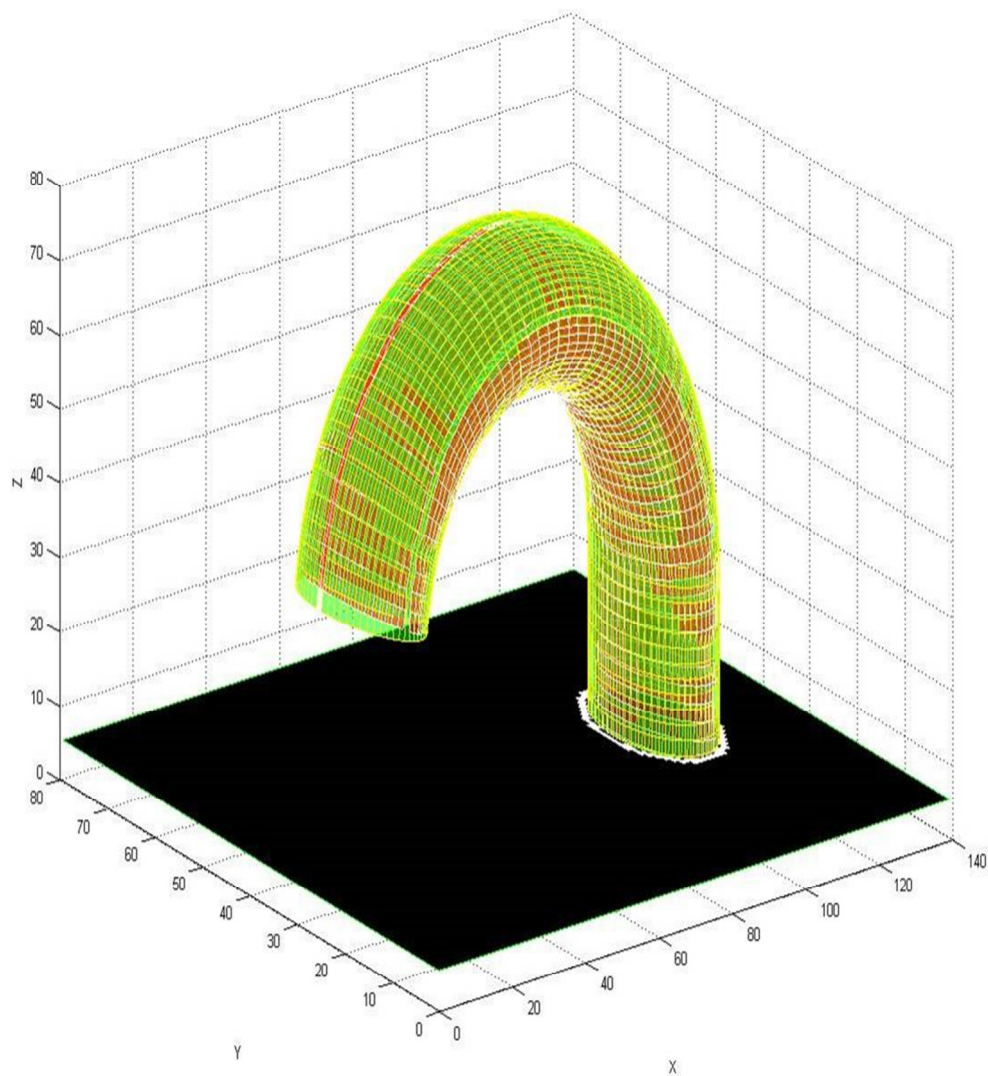


Figure 6.2: Phantom (green) and reconstructed (red) meshes embedded in 3-D dataset for the synthetic aorta. Units in mm.

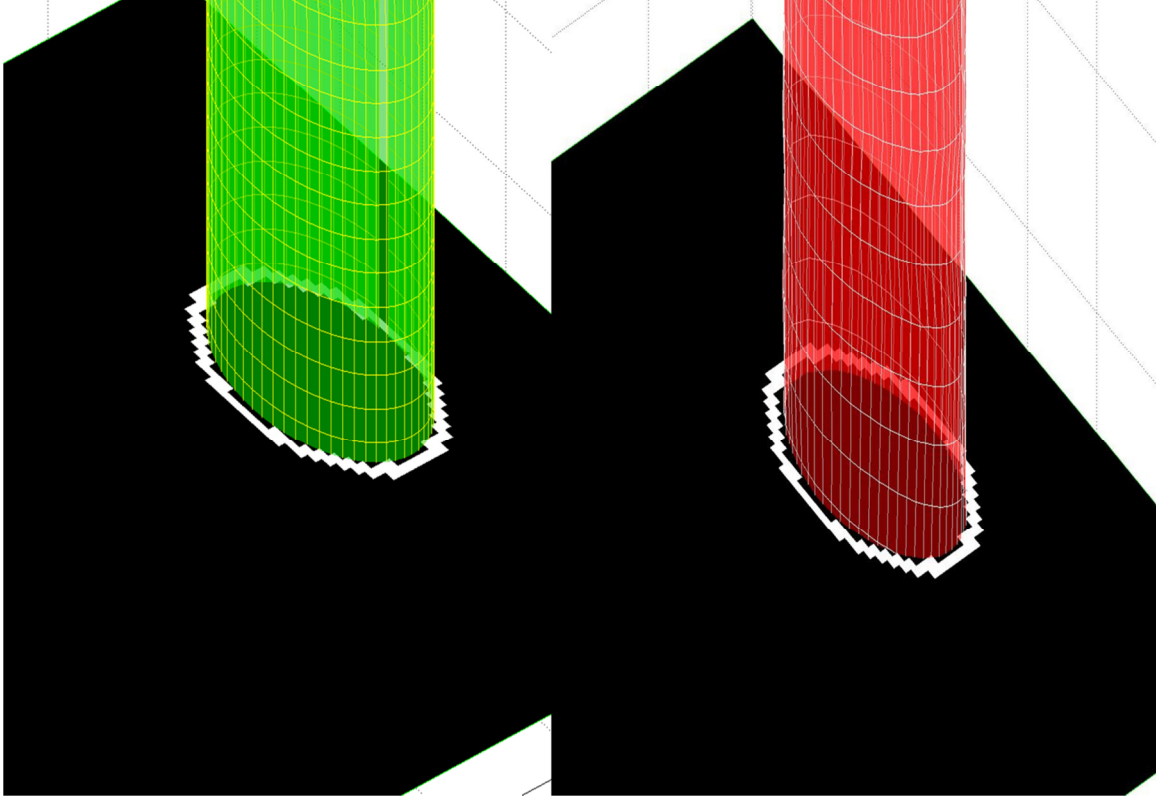


Figure 6.3 left: Phantom mesh, right: reconstructed mesh.

Then, the reconstruction accuracy was assessed by measuring, for each cross-section, how far nodes of the phantom mesh would be from nodes of the reconstructed mesh, using a normalized mean square error (NMSE) and bias defined as in (Lu et al., 2009):

$$\%NMSE = 100 * \left[\frac{\sum_{i=1}^n (x_i - p_i)^2}{\sum_{i=1}^n p_i^2} \right] \quad (6.1)$$

$$\%bias = 100 * \left[\frac{\sum_{i=1}^n x_i - \sum_{i=1}^n p_i}{\sum_{i=1}^n p_i} \right] \quad (6.2)$$

where i is the node index in the mesh and x and p are the spatial coordinates of the reconstructed and phantom meshes, respectively.

The first and last reconstructed cross-sections were omitted because they are susceptible to artefacts. Therefore, 58 out of 60 cross-sections were considered, with 60 nodes each.

The results for NSME and bias values are shown in Table 6.1

Table 6.1 Quantitative evaluation of mesh reconstruction accuracy for the synthetic aorta, for 58 cross-sections and 60 nodes per cross-section.

	Mean value	Maximum value	Minimum value	Standard Deviation
NMSE (%)	1.03	2.09	0.01	0.73
Bias (%)	0.07	0.14	-0.14	0.10

The average values for NSME and bias were 1.03% and 0.07%, respectively, which represents a high accuracy of reconstruction for the proposed methodology, also confirmed by the low value of the NSME standard deviation.

6.2 Mesh quality inspection

The quality of the mesh is a significant factor in the accuracy and stability of ulterior numerical computations. Mesh quality as defined by (Knupp, 2007) “is concerned with characteristics of a mesh that permits a particular numerical PDE simulation to be efficiently performed to the underlying physics and with the accuracy required for the problem”. Consequently, regardless of the type of mesh used, checking the grid quality is essential. Depending on the cell types in the mesh (tetrahedral,

hexahedral, quad, etc.), different quality criteria are evaluated. In the case at hand, since the generated surface meshes are meant to describe shell elements, the following quality factors were investigated:

- **Aspect ratio:**

The aspect ratio shows the maximum dimension ratio between opposing edges, faces or principal directions in an element. For example, in a quadrilateral element, the aspect ratio represents the ratio of length to width. Normally, finite element simulations are more accurate when the aspect ratio is closest to 1.

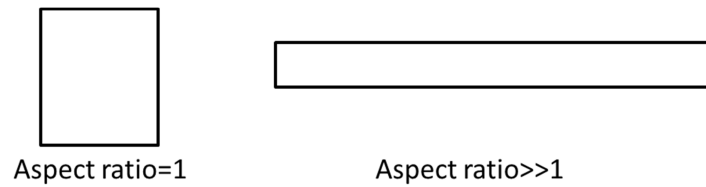


Figure 6.4: Aspect ratio of a quadrilateral element.

- **Warpage**

Warpage measures the degree to which a quadrilateral element's corner points are out of plane from the centroid plane of the element. Warpage happens when the 4 nodes of the element are not in plane.

- **Skewness**

Skewness measures the angular deviation from a rectangular shape in surface elements. It is defined for a quadrilateral element as:

$$\max\left(\frac{\theta_{max}-90}{90}, \frac{90-\theta_{min}}{90}\right) \quad (6.3)$$

where the angles are described in Figure 6.5.

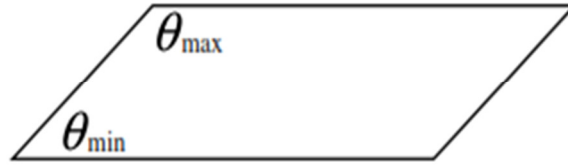


Figure 6.5: Skewness angles for a quadrilateral element.

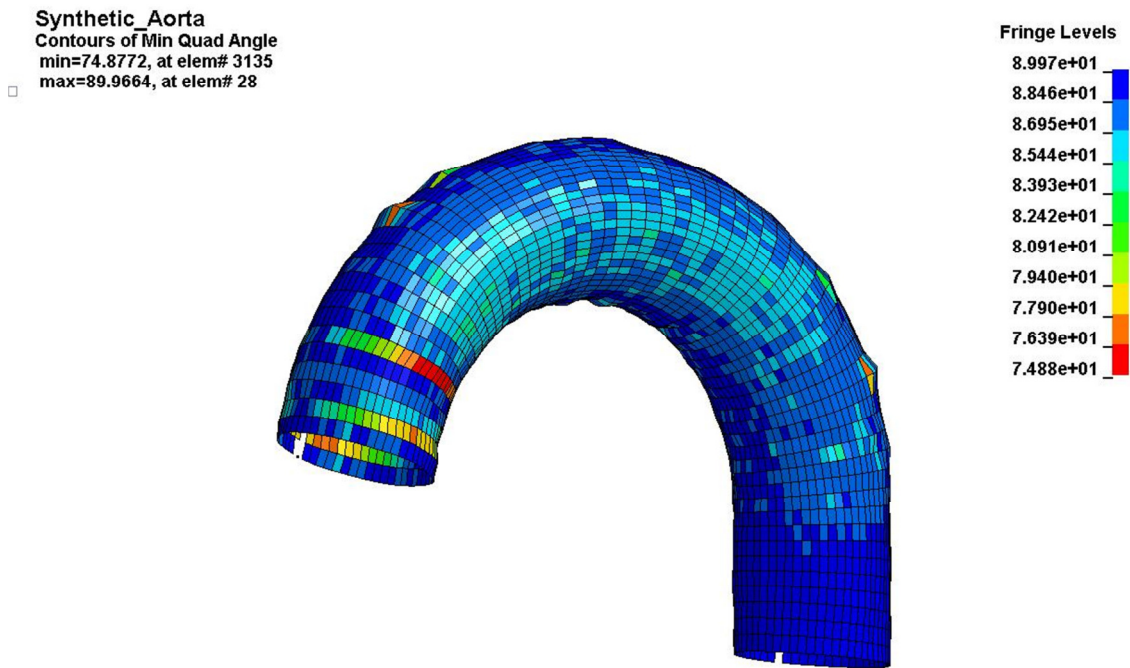
- Jacobian ratio

The Jacobian ratio characterizes the deviation of the element from an assumed ideal shape. The ideal shape for an element depends on the element type. The check is performed by mapping an ideal element in parametric coordinates onto the actual element defined in global coordinates. For example, the coordinates of the corners of an ideal quadrilateral element in parametric coordinates are (-1,-1), (1,-1), (1,1), and (-1,1). The Jacobian ratio describes the local stretching of the parametric space required to fit it onto global coordinate space (Dhole, 2010).

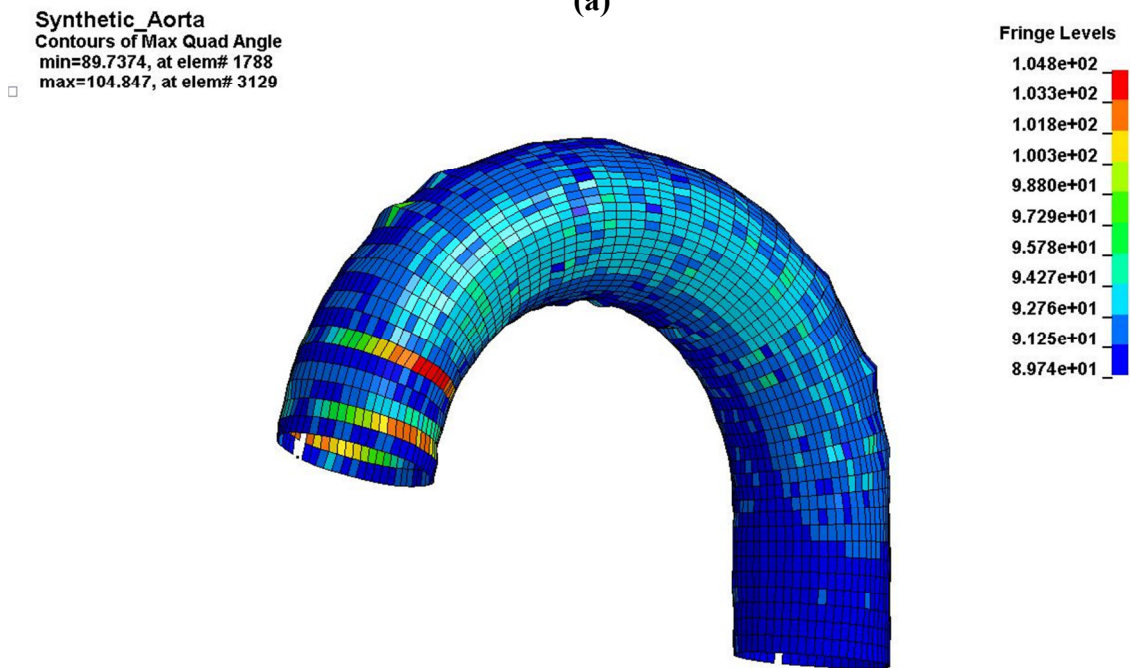
The Jacobian ratio values range from -1.0 to 1.0, where 1.0 represents a perfectly shaped element. The minimum acceptable values range from 0.6 to 0.2 (Mahmoudi, 2013). It has been suggested that elements with negative scaled Jacobian ratios should not be recommended for analysis and should be removed (Shepherd et al., 2007).

As an illustration, LS-DYNA provides users with a tool for mesh quality analysis which includes the criteria; the individual element results are conveniently shown with different colors and the maximum and minimum values are noted as well. The results are

illustrated below for the synthetic aorta (Figure 6.6) and the Visible Human datasets (Figure 6.7).

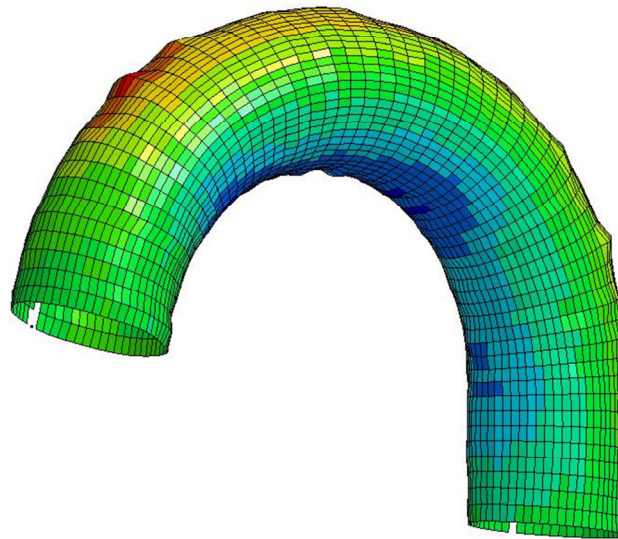


(a)



(b)

Synthetic_Aorta
Contours of Aspect Ratio
min=1.17532, at elem# 2791
max=3.11272, at elem# 2702

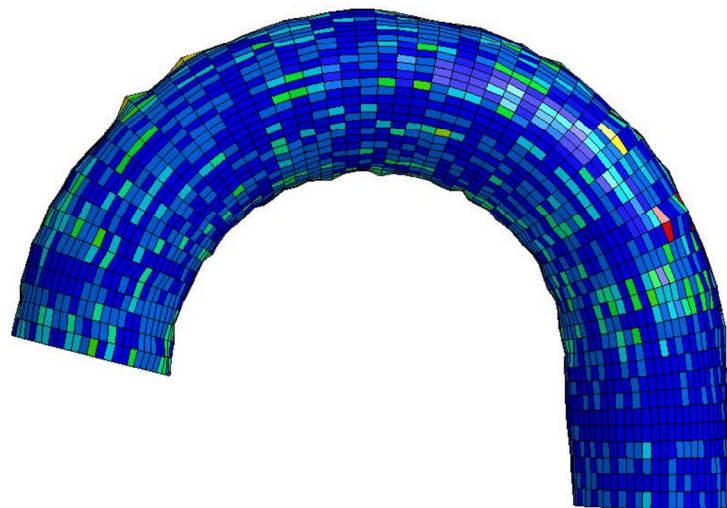


Fringe Levels

3.113e+00
2.919e+00
2.725e+00
2.532e+00
2.338e+00
2.144e+00
1.950e+00
1.757e+00
1.563e+00
1.369e+00
1.175e+00

(c)

Synthetic_Aorta
Contours of Warpage
min=0, at elem# 3
max=38.8859, at elem# 982

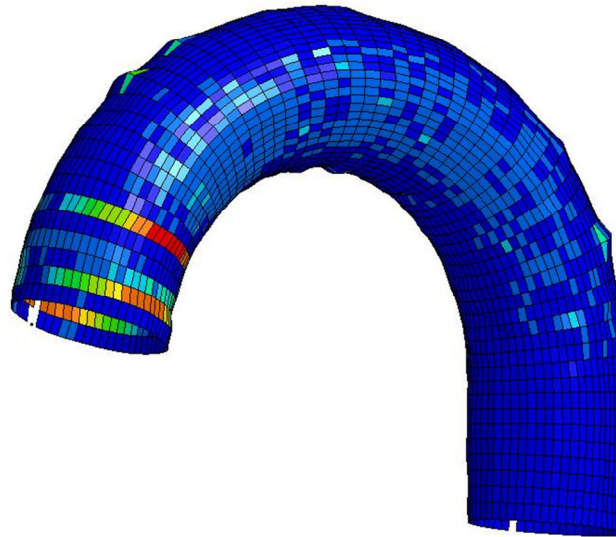


Fringe Levels

3.889e+01
3.500e+01
3.111e+01
2.722e+01
2.333e+01
1.944e+01
1.555e+01
1.167e+01
7.777e+00
3.889e+00
0.000e+00

(d)

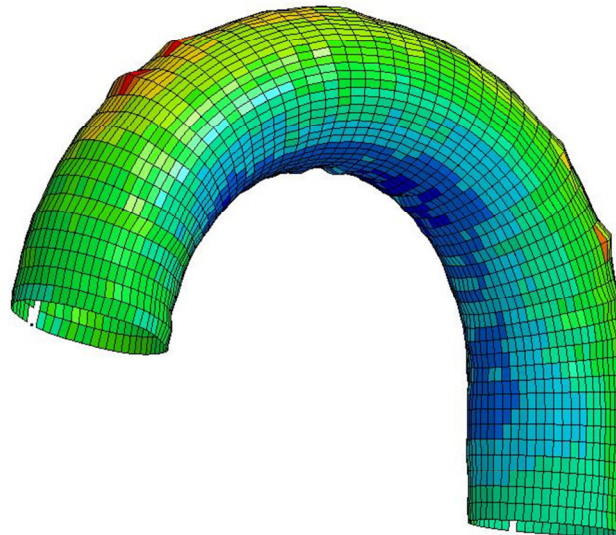
Synthetic_Aorta
Contours of Skew
min=0, at elem# 8
max=13.6096, at elem# 3155



Fringe Levels
1.361e+01
1.225e+01
1.089e+01
9.527e+00
8.166e+00
6.805e+00
5.444e+00
4.083e+00
2.722e+00
1.361e+00
0.000e+00

(e)

Synthetic_Aorta
Contours of Jacobian
min=0.70993, at elem# 2702
max=0.990741, at elem# 2791



Fringe Levels
9.907e-01
9.627e-01
9.346e-01
9.065e-01
8.784e-01
8.503e-01
8.223e-01
7.942e-01
7.661e-01
7.380e-01
7.099e-01

(f)

Figure 6.6: Mesh quality inspection for the reconstructed synthetic aorta with 3480 elements, (a): minimum quad angles, (b): minimum quad angles, (c): Aspect ratio, (d): Warp, (e): Skewness, (f): Jacobian ratio.

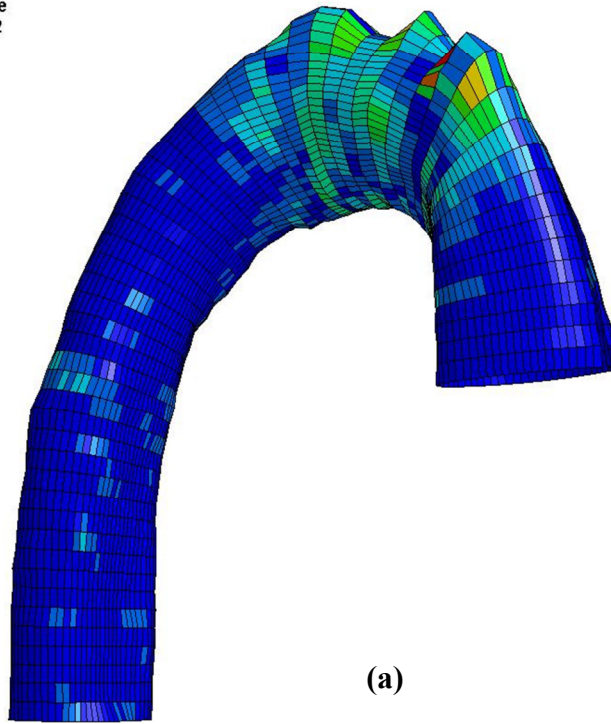
Table 6.2 lists the maximum and minimum values of the above criteria in addition to their comparison with the standard allowable pre-set values in LS-DYNA. The results for the mesh are generally good according to all criteria and except for one, fit in the allowable range. For the warpage criteria, 616 out of 3480 elements exceed the LS-DYNA pre-set allowable range. Since warpage relates to non-planar elements, one solution to reduce the number of elements exceeding the warpage limit would be to increase the number of elements. On the other hand, the admissible values could be set differently depending on the usage of the mesh.

Table 6.2 Mesh quality summary for the synthetic aorta.

	Allowable	Minimum value	Maximum value	# of violated
Minimum quad angle	45	74.9	90	0(0%)
Maximum quad angle	135	89.7	105	0(0%)
Aspect ratio	10	1.18	3.11	0(0%)
Warpage	10	0	38.9	616(17.7%)
Skewness	45	0	13.6	0(0%)
Jacobian ratio	0.6	0.71	0.991	0(0%)

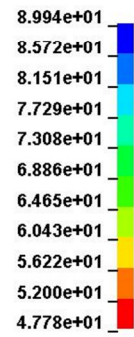
Figure 6.7 and Table 6.3 similarly present results from the quality inspection tests for Visible Human dataset.

VisibleHuman
Contours of Min Quad Angle
min=47.7846, at elem# 2602
max=89.9378, at elem# 9

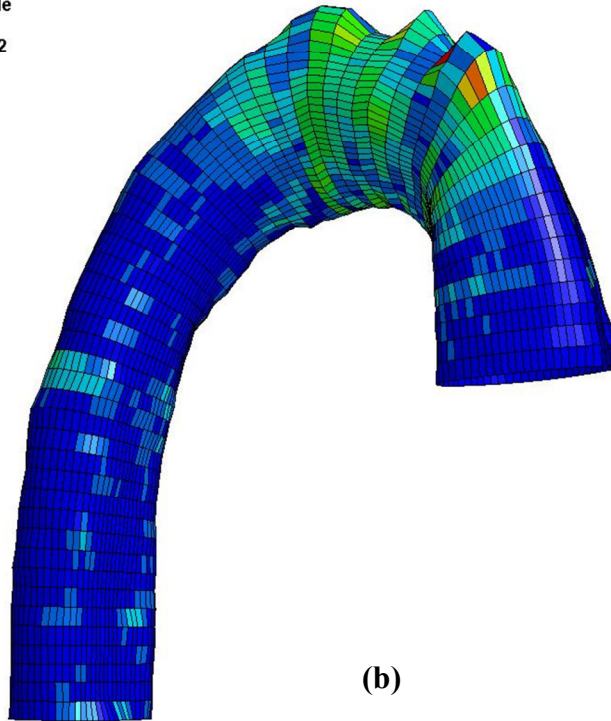


(a)

Fringe Levels

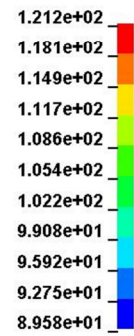


VisibleHuman
Contours of Max Quad Angle
min=89.5846, at elem# 155
max=121.248, at elem# 2602



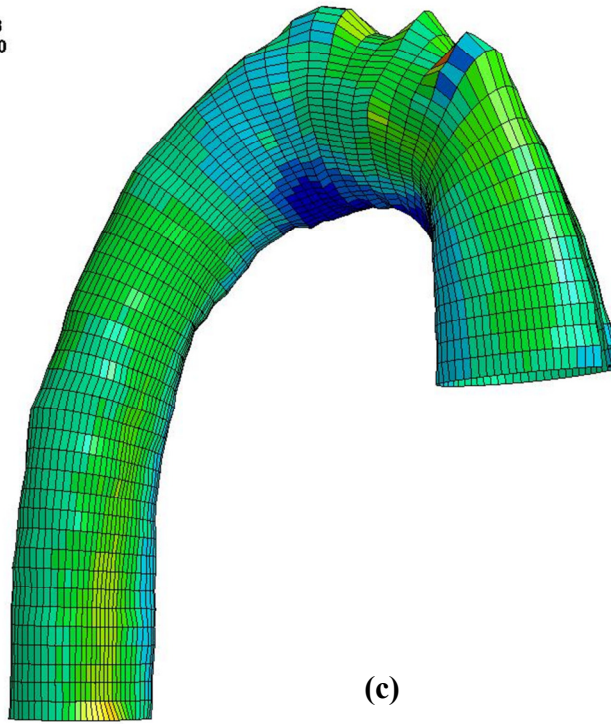
(b)

Fringe Levels



VisibleHuman
Contours of Aspect Ratio
min=1.07286, at elem# 3043
max=5.34577, at elem# 2600

□



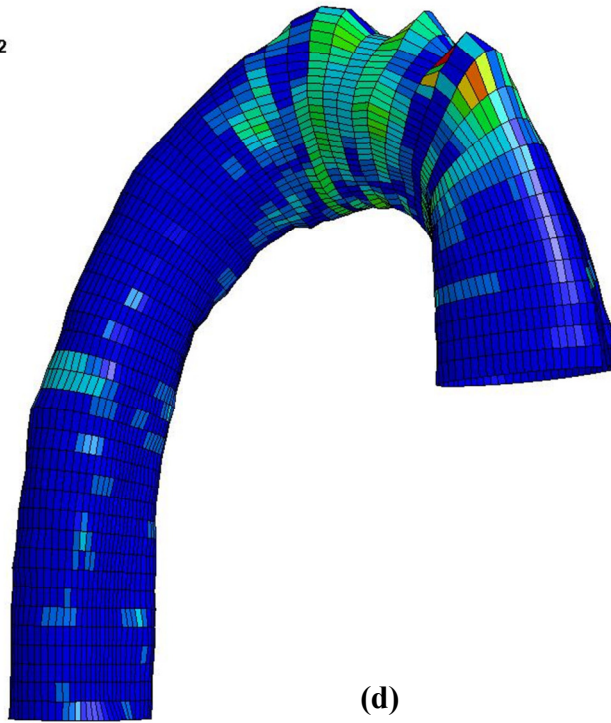
(c)

Fringe Levels



VisibleHuman
Contours of Skew
min=0, at elem# 9
max=29.4346, at elem# 2602

□



(d)

Fringe Levels



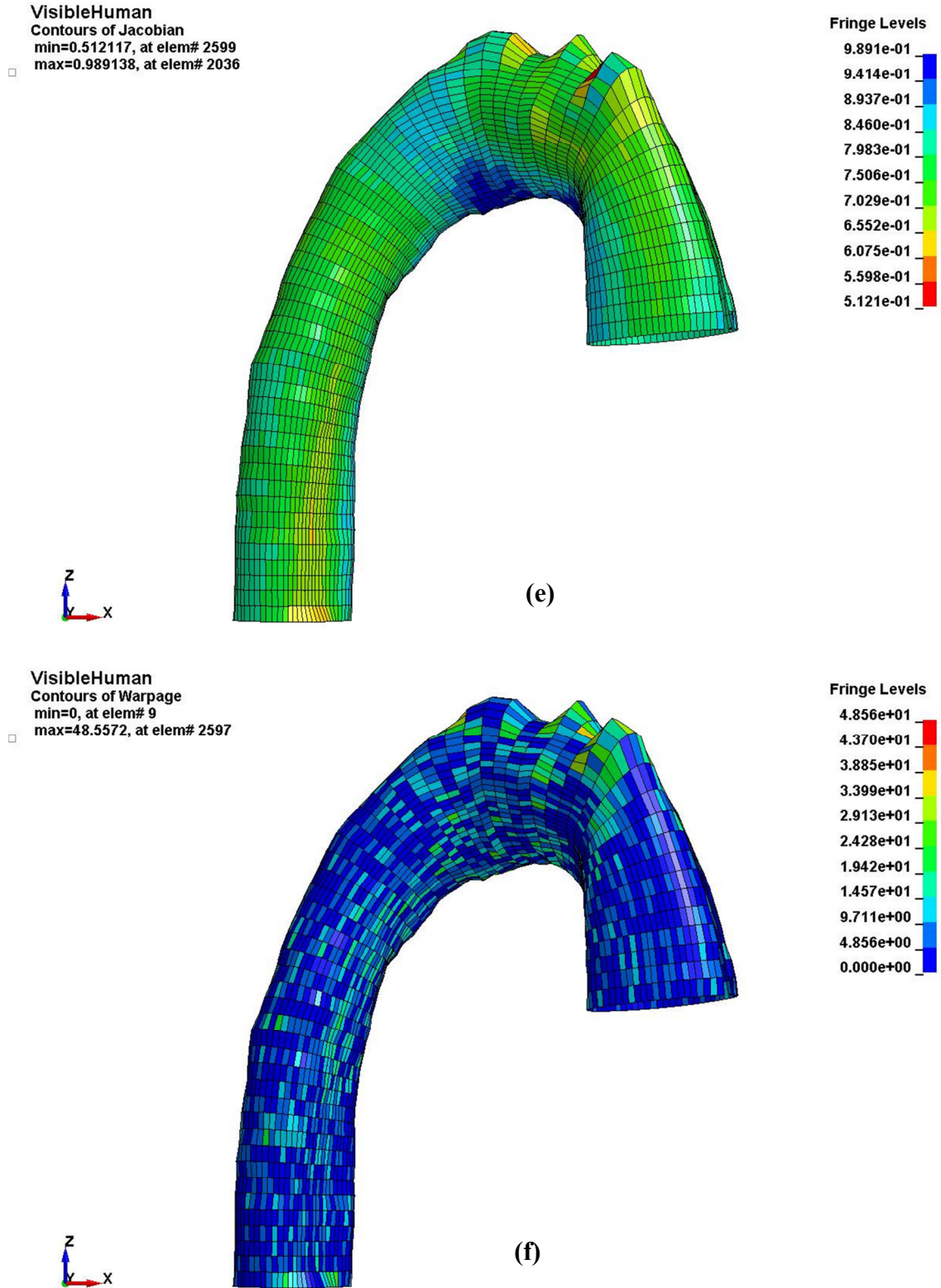


Figure 6.7: Mesh quality inspection for the reconstructed Visible Human dataset, (a): minimum quad angles, (b): minimum quad angles, (c): Aspect ratio, (d): Warpage, (e): Skewness, (f): Jacobian ratio.

Table 6.3: Mesh quality summary for the Visible Human data set.

	Allowable	Minimum value	Maximum value	# of violated
Minimum quad angle	45	47.7	89.9	0 (0%)
Maximum quad angle	135	89.6	121	0 (0%)
Aspect ratio	10	1.07	5.35	0 (0%)
Warpage	10	0	48.6	672 (19.3%)
Skewness	45	0	29.4	0 (0%)
Jacobian ratio	0.6	0.51	0.98	0 (0%)

Again, for the Visible Human dataset, the mesh satisfies all criteria, except for warpage criteria, for which 672 out of 3480 elements exceed the LS-DYNA pre-set allowable range. As mentioned previously, one solution to reduce the number of elements exceeding the warpage limit would be to increase the number of elements.

Mesh quality inspection for the meshes reconstructed from the other datasets considered in this work were similar to the previous one and therefore, are not shown.

In finite element analyses, one source of error usually include too coarse meshes, high skewness and non-conforming adjacent cells (Bakker, 2008). Mesh coarseness must be evaluated on a case by case basis through mesh sensitivity analysis. Non-conforming elements do not exist in structured meshes. The results in Tables 6.2 and 6.3 also tell us that high-skewness is not an issue. To conclude, these observations suggest that the proposed methodology is not only fast, it also delivers structured meshes of good quality.

7 Volume mesh generation

The purpose of the present chapter is to describe how the methodology used to obtain a structured surface mesh of the aortic inner wall can be expanded to generate volume meshes for the aortic wall and/or the blood in the lumen. As will be explained, these meshes fundamentally rely on the construction of two additional surface meshes, one being the surface mesh of the outermost aortic wall; the other being the surface mesh of an arbitrarily-shaped contour embedded in the fluid region. Before addressing the fluid mesh, let us present the ideas underpinning the construction of a structured volume mesh for the aortic wall.

7.1 Aortic wall volume mesh generation

Histologically, the aortic wall can be differentiated into three different layers: from the lumen to the outside, a thin inner layer called intima; a thick, elastic middle layer called the media, and a thin outer layer called the adventitia (Silver et al., 1989). However, for demonstration of principle, the aortic wall will be considered in the following as one sheet that can be discretized into an arbitrary number of layers. In later applications, these layers may be assigned different material properties as dictated by histology.

The main idea used to construct the surface mesh of the outermost aortic wall comes from the observation that, in vivo, the thickness of blood vessels, including the aorta, is on average $1/10$ of the local radius of the vessel (McDonald, 1974). As the centerline and local radius information has already been collected during the construction

of the structured mesh of the aortic lumen, it is straightforward to generate a second structured surface mesh where the local radius for each constitutive node is 10% larger than that for the nodes in the previous mesh. In the absence of thrombus (as discussed in Section 5.11), this second layer will represent the outermost aortic wall, as illustrated in Figures 7.1 to 7.3. More structured surface meshes may be easily sandwiched between these two fundamental layers, and then hexahedral elements formed between two adjacent layers by joining quadrilaterals that face each other. This will result in a refined structured volume mesh of the aortic wall.

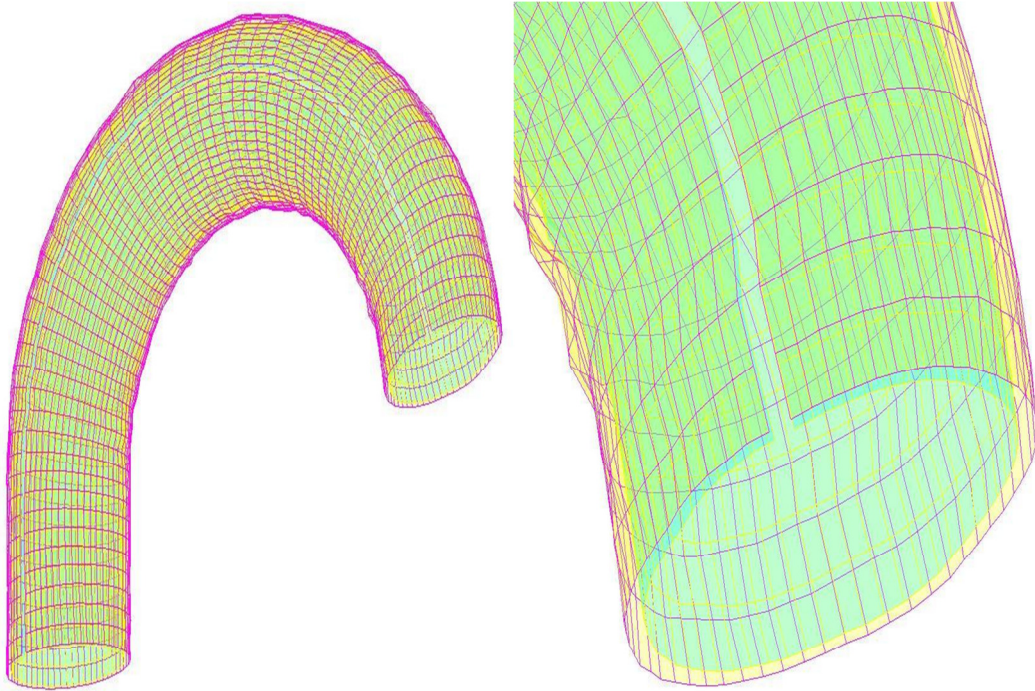


Figure 7.1: Surface meshes of the outer wall (magenta wireframe, yellow shade) and inner wall (yellow wireframe, cyan shade) to be used toward hexahedral meshing of the aortic wall in the synthetic aorta dataset.

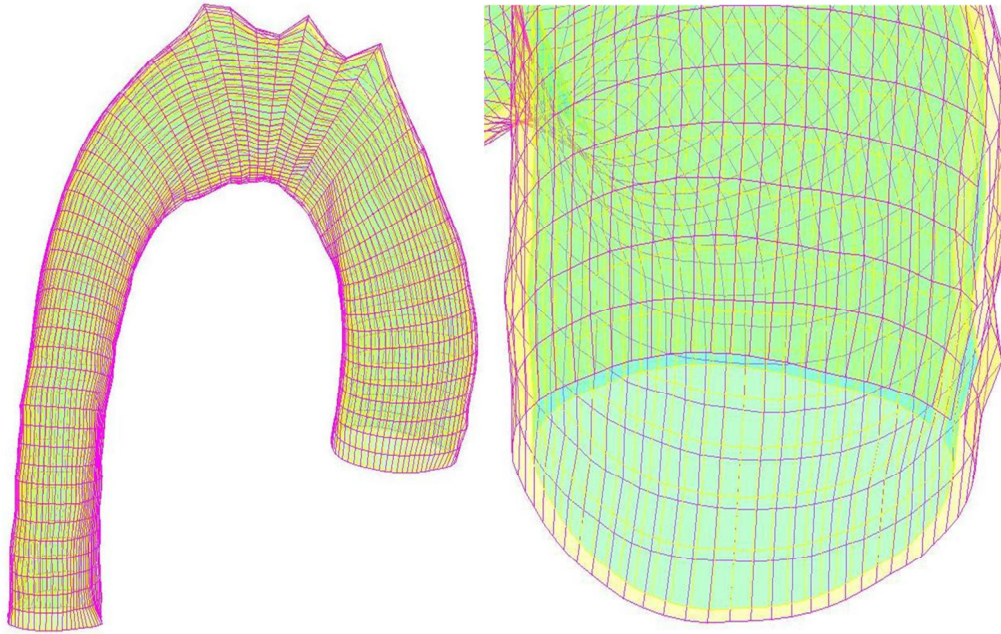


Figure 7.2: Surface meshes of the outer wall (magenta wireframe, yellow shade) and inner wall (yellow wireframe, cyan shade) to be used toward hexahedral meshing of the aortic wall in the Visible Human dataset.

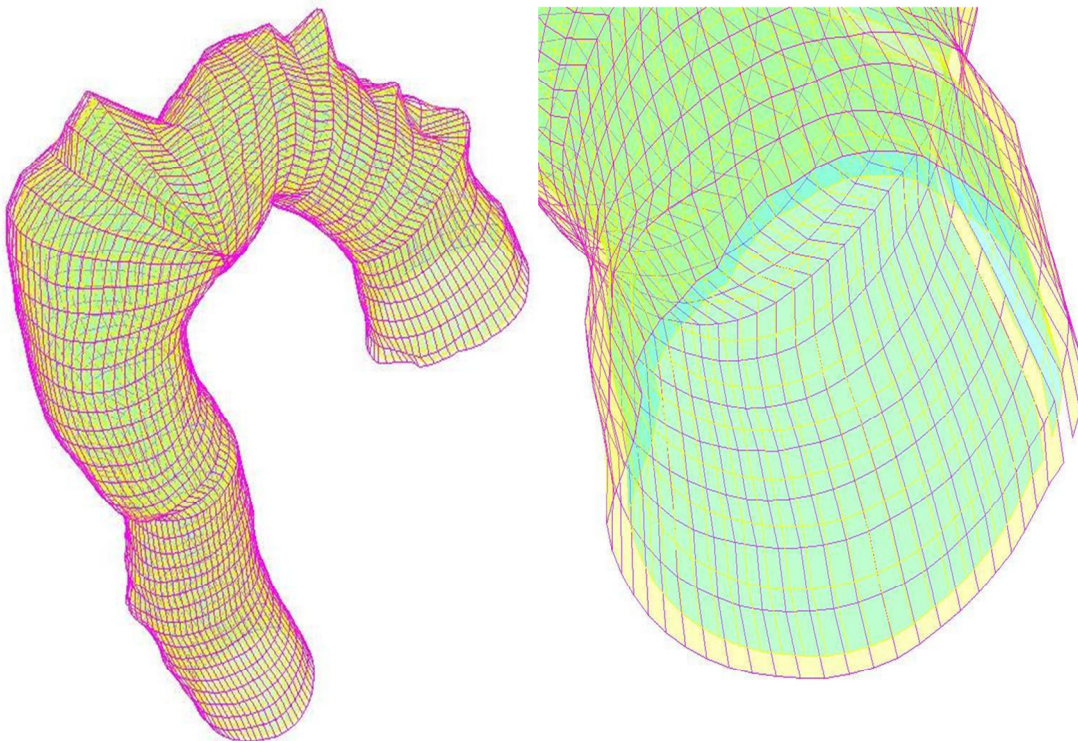


Figure 7.3: Surface meshes of the outer wall (magenta wireframe, yellow shade) and inner wall (yellow wireframe, cyan shade) to be used toward hexahedral meshing of the aortic wall in the descending aortic aneurysm dataset.

7.2 Fluid domain volume mesh

In this section, the goal is to create a structured mesh of the fluid domain inside the aortic lumen. Again, we want to demonstrate that the centerline and local radius information collected during the construction of the structured surface mesh of the aortic lumen can be used toward this goal. The first step is the realization that a hexahedral mesh of the cylindrical fluid domain cannot be obtained by simply moving node layers radially toward the centerline. Instead, a fluid core tube of arbitrary shape can be defined such that it can easily be meshed with hexahedral elements and mapped to the structured surface mesh of the lumen.

In what follows a core tube of square cross-section is defined and centered on the aortic centerline. In each aortic cross-section as defined for the structured mesh of the lumen, $ncirc$ nodes on the perimeter of the tube are generated using the algorithm presented in Figure 7.4. The size of the square is arbitrarily chosen as 75% of the radius of the current cross-section. The resulting core is shown in Figure 7.5.

```

Inputs: Number of nodes along circumference (ncirc)
           R (generated radius form Ray-tracing)
           Center (Centroids generated form Ray-tracing)

Output: X(x, y, z) quad (Coordinates of the core for fluid mesh)

BEGIN
Initialize R as .75* min (R)
s=equally spaced points  $\in [\frac{2\pi}{ncirc}, 2\pi]$ 
FOR i = 1:ncirc

    IF  $s_i > 0$  AND  $s_i = \frac{\pi}{2}$ 
         $r_i = R/(\sqrt{2} * \sin(s + \frac{\pi}{4}))$ 
    ELSEIF  $s_i \geq \frac{\pi}{2}$  AND  $s_i \leq \pi$ 
         $r_i = R/(\sqrt{2} * \sin(s - \frac{\pi}{4}))$ 
    ELSEIF  $s_i > \pi$  AND  $s_i \leq 3\frac{\pi}{2}$ 
         $r_i = -R/(\sqrt{2} * \sin(s + \frac{\pi}{4}))$ 
    ELSEIF  $s_i > 3\frac{\pi}{2}$  AND  $s_i \geq 2\pi$ 
         $r_i = -R/(\sqrt{2} * \sin(s - \frac{\pi}{4}))$ 
    ENDIF

ENDFOR

X(x, y, z)quad = Center +  $r_i * (\cos(s) * x + \sin(s) * y)$ 
END

```

Figure 7.4: Algorithm for creation of square core tube mesh inside fluid domain.

More structured surface meshes may be easily sandwiched between the aortic inner wall surface mesh and the core tube (Figure 7.6). In doing so, more layers may be defined close to the aortic wall and fewer toward the core, so that fluid boundary layers may be resolved efficiently during CFD analyses. Then, hexahedral elements can be

formed between two adjacent layers by joining quadrilaterals that face each other. The core tube can be directly meshed with hexahedral elements (Figure 7.6). This will result in a refined structured volume mesh of the fluid domain.

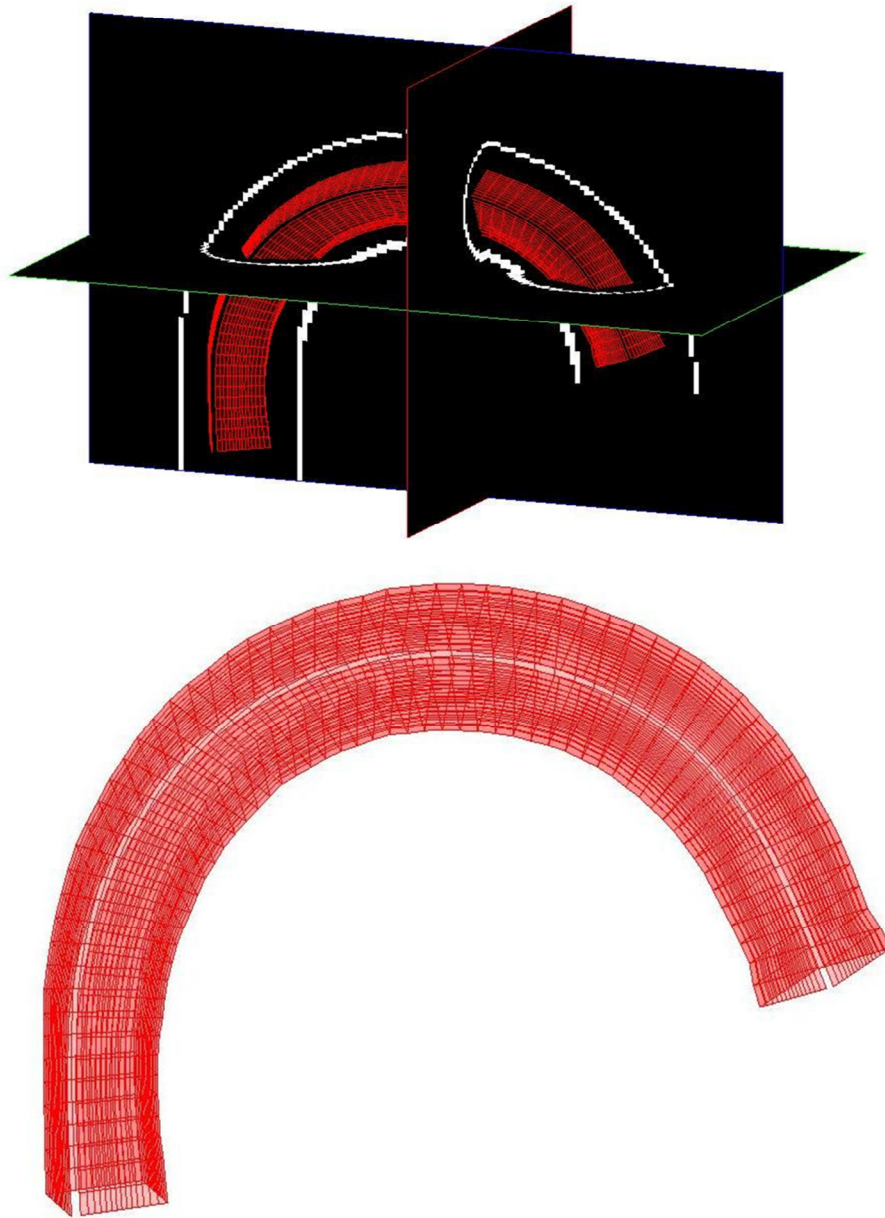


Figure 7.5: Core surface mesh inside fluid domain of synthetic aorta.

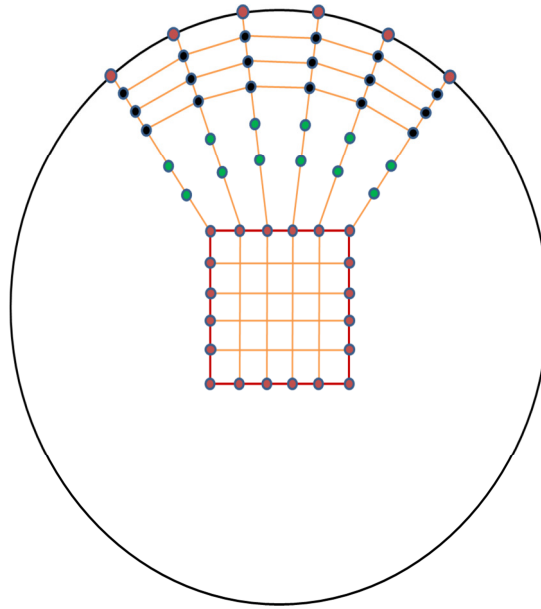


Figure 7.6: Node pattern for hexahedral meshing of the fluid domain, based on the surface mesh of the inner aortic wall and the square core tube. The nodes in black may be used to generate a refined mesh near the aortic wall, while those in green and in the square core may define a coarser mesh.

Figures 7.7 to 7.9 show the three fundamental surface layers to be used to generate the actual hexahedral volume meshes for the aortic wall, the fluid domain or both. The red surface mesh is that of the square fluid core, the magenta surface mesh is that of the outer aortic wall while the yellow surface mesh is that of the lumen or inner aortic wall. Since the generated volume meshes rely on the surface mesh of the aortic inner wall only, their generation takes less than 10 seconds. These results, albeit still preliminary, demonstrate the feasibility of the structured volume mesh generation. Once created, the volume mesh information will be exported to third-party software by creating a node list and connectivity tables, similarly to what was described in Section 5.11.

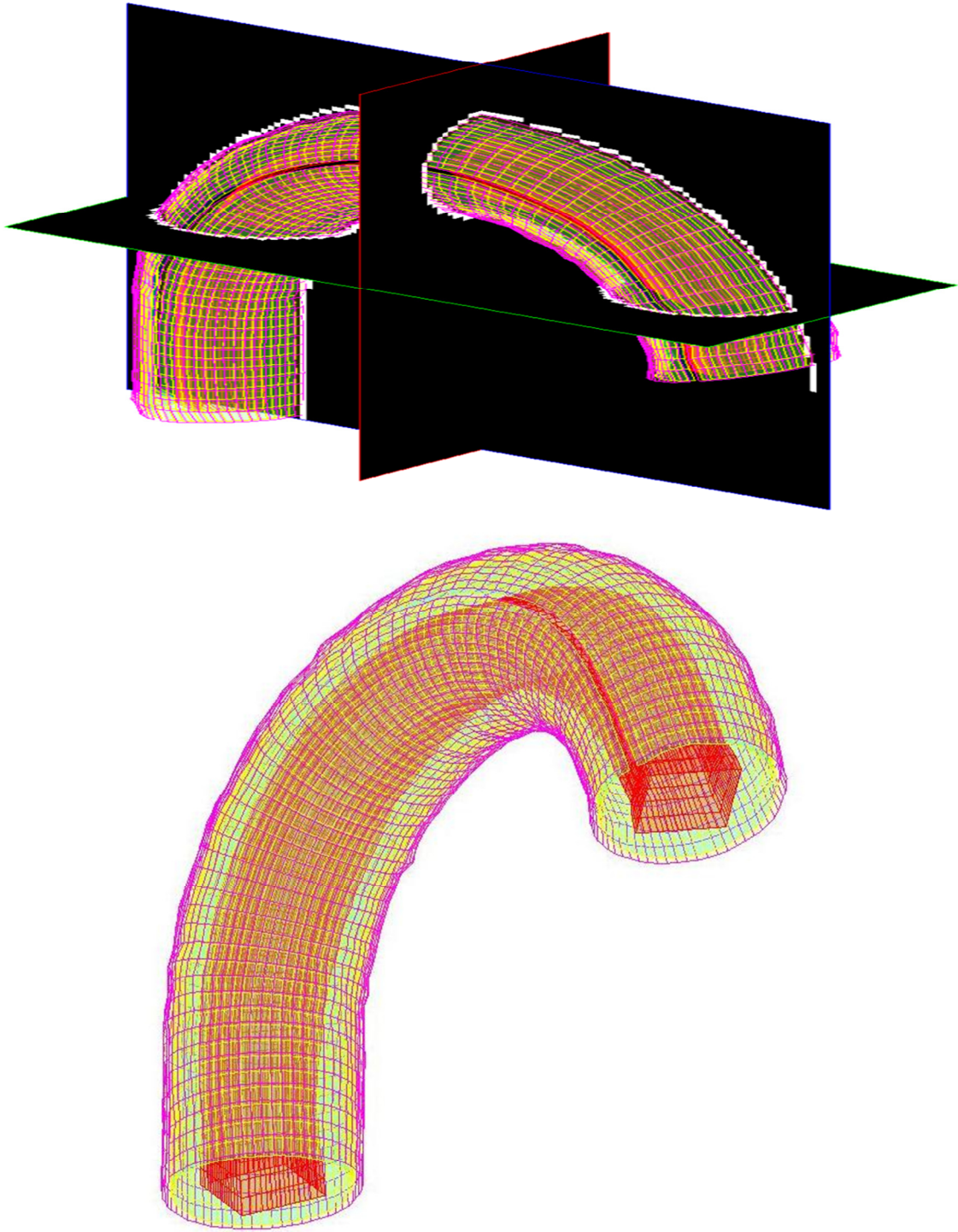


Figure 7.7: Three fundamental surface meshes for the synthetic aorta dataset.

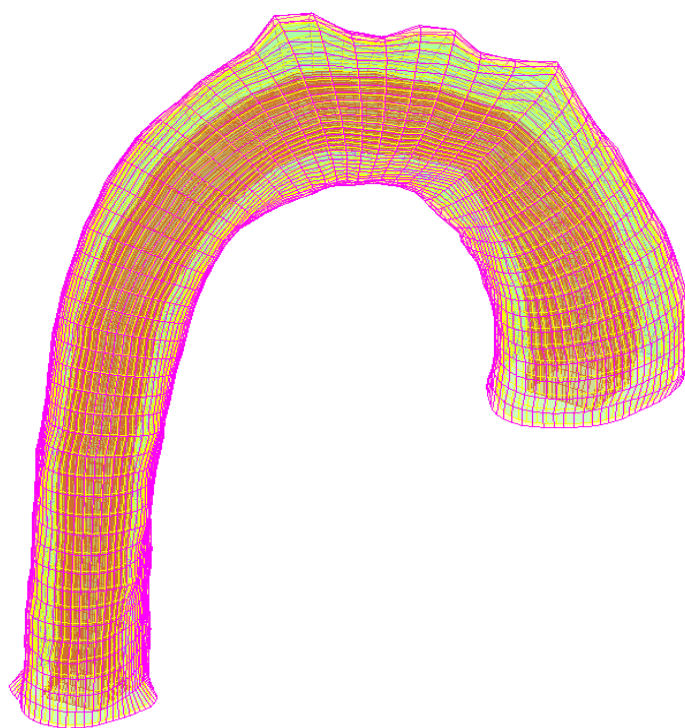
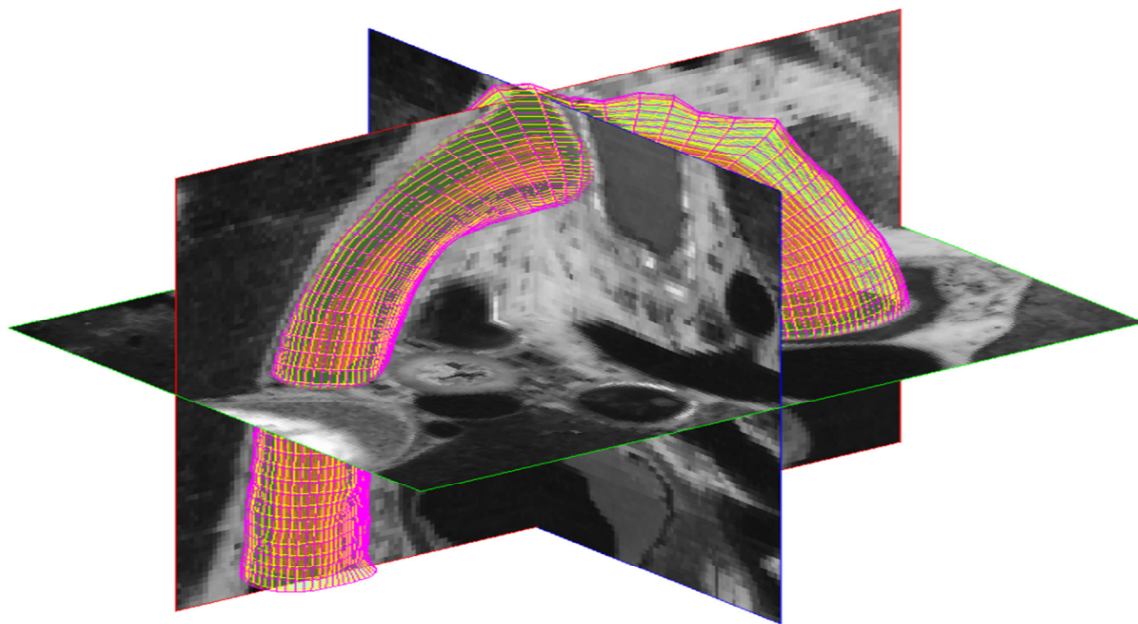


Figure 7.8: Three fundamental surface meshes for the Visible Human dataset.

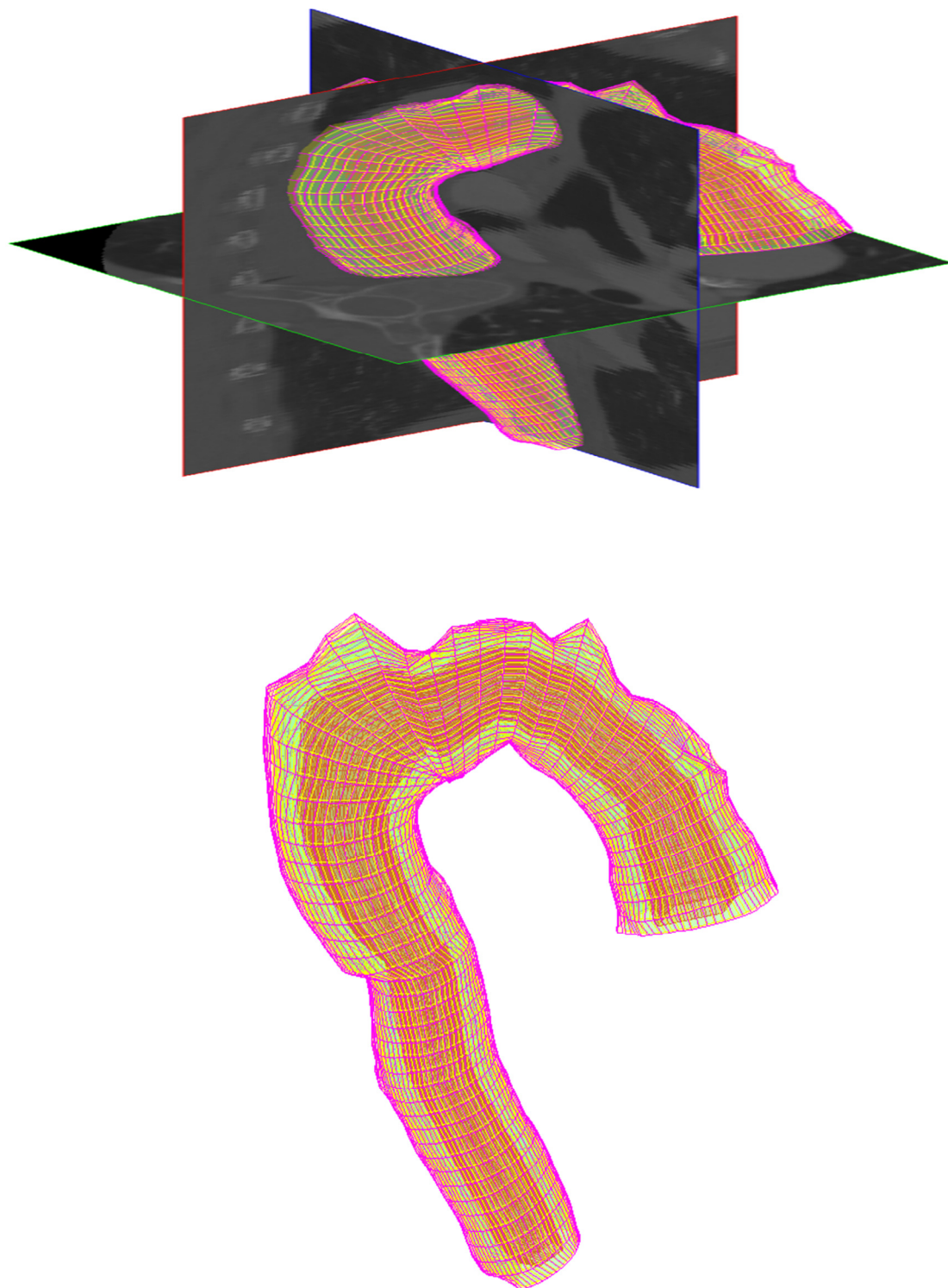


Figure 7.9: Three fundamental surface meshes for the descending aortic aneurysm dataset.

8 Mesh merging for aortic branches

One sizeable issue that remains to be addressed pertains to aortic branches. Three major branches of the aorta are normally present at the top of the aortic arch, namely the brachiocephalic trunk, the left common carotid artery, and the left subclavian artery (recall Figure 2.1). From a structural point of view, branches, and especially where they connect with the main vessel of the aorta, are locations where mechanical stress concentrations occur because of abrupt changes in geometry. From a CFD point of view, branches are flow dividers that potentially incur vortices, flow reattachment regions and high fluid shear stress, especially considering the pulsatile nature of the blood flow. It is firmly held in the cardiovascular mechanics research community that for either or both reasons, branches are prime targets for atherosclerotic developments and as such, deserve attention in modeling. Therefore, if a new viable meshing technology is to be proposed for the aorta, it should include effective ways to handle branches.

In this chapter, branches of the aorta are first meshed separately using the methodology described previously. Then, the transition regions where the branches connect to the main aorta are determined. As a crude demonstration of principle, these regions are first decomposed using bilinear patches to merge meshes from different vessels. Then, higher order patches will be discussed for their capability to generate smoother transition regions. Because of time constraints placed on the research project, the approach is admittedly still exploratory in nature and the results more preliminary than in the previous chapters. In particular, the illustrations below are only based on the

CT dataset of a thoracic aortic aneurysm, and as a first step, only surface meshing of the inner aortic wall is considered.

8.1 Individual branch meshing

With the main aorta already meshed, each branch can be meshed sequentially using the same methodology as described in Chapter 5. Briefly, the user defines the initial approximate center point and direction of mesh generation for a branch. The direction considered for all branches was from top to bottom as it was easier to start where the branch cross-sections were mostly circular, i.e. away from the region of branching. The mesh reconstruction for each branch was stopped when getting close to the aortic arch, where, typically, the reconstruction procedure would fail because of highly disturbed 3-D GVF field. For each branch, the mesh generation time was approximately 10 seconds. The process is illustrated in Figure 8.1.



Figure 8.1: Top to bottom: branch meshing process; Left: left subclavian artery, Middle: left common coronary artery, Right: brachiocephalic trunk.

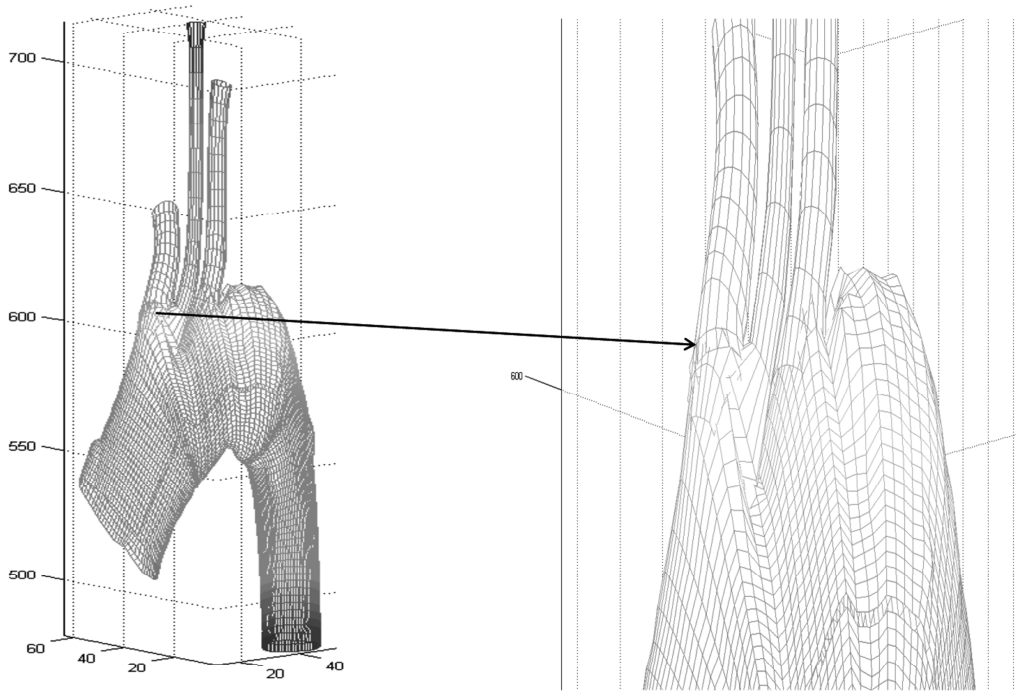


Figure 8.2: Disjointed initial meshes of aorta and branches.

Figure 8.2 presents the four disjointed meshes obtained for the aorta and its three main branches in the aortic arch. The next issue is to connect these meshes together to make them one single structured mesh.

8.2 Bilinear patches and decomposition of transition region

A useful concept toward structured mesh merging is that of bilinear patches. A bilinear patch is formed as a combination of four possibly non-coplanar corner points $(P_{00}, P_{01}, P_{10}, P_{11})$. Any point P in the patch is represented as a bilinear function of the four corner points such that (Ramsey et al., 2012):

$$P(u, v) = (1 - u)(1 - v)P_{00} + (1 - u)vP_{01} + u(1 - v)P_{10} + uvP_{11}, \quad (8.1)$$

where $u, v \in [0,1]$.

From linearity, the edges of a bilinear patch are straight lines between corner points. Nodes for a structured mesh based on a bilinear patch can easily be generated by taking discrete equally spaced values for parameters u and v .

The basic idea behind mesh merging as applied herein simply consists in decomposing the transition regions around the roots of the aortic branches into sets of bilinear patches. The first step is to identify the transition regions in the aortic arch and remove mesh cross-sections associated with them. This is done for the left subclavian artery in Figure 8.3. Cross-sections $\Gamma 1$ and $\Gamma 2$ delimit the transition region toward the left subclavian artery in the main aortic arch. The choice of the proximal cross-section of the branch (i.e. closest to the aorta) to be retained for mesh merging is up to the user.

When this choice is made, the decomposition of the transition region begins with determining points A and B, respectively defined as the points in the proximal cross-section of the branch that are at minimum distance from cross-sections $\Gamma 1$ and $\Gamma 2$. The points on these cross-sections that are closest to A and B are labelled C and D, respectively, in Figure 8.3. Between A and C, the user arbitrarily adds point E such that a cubic spline drawn through AEC is a smooth edge transitioning from the aortic arch to the branch. Similarly, Point F is added between points B and D, and a smooth transition spline is drawn through BFD.

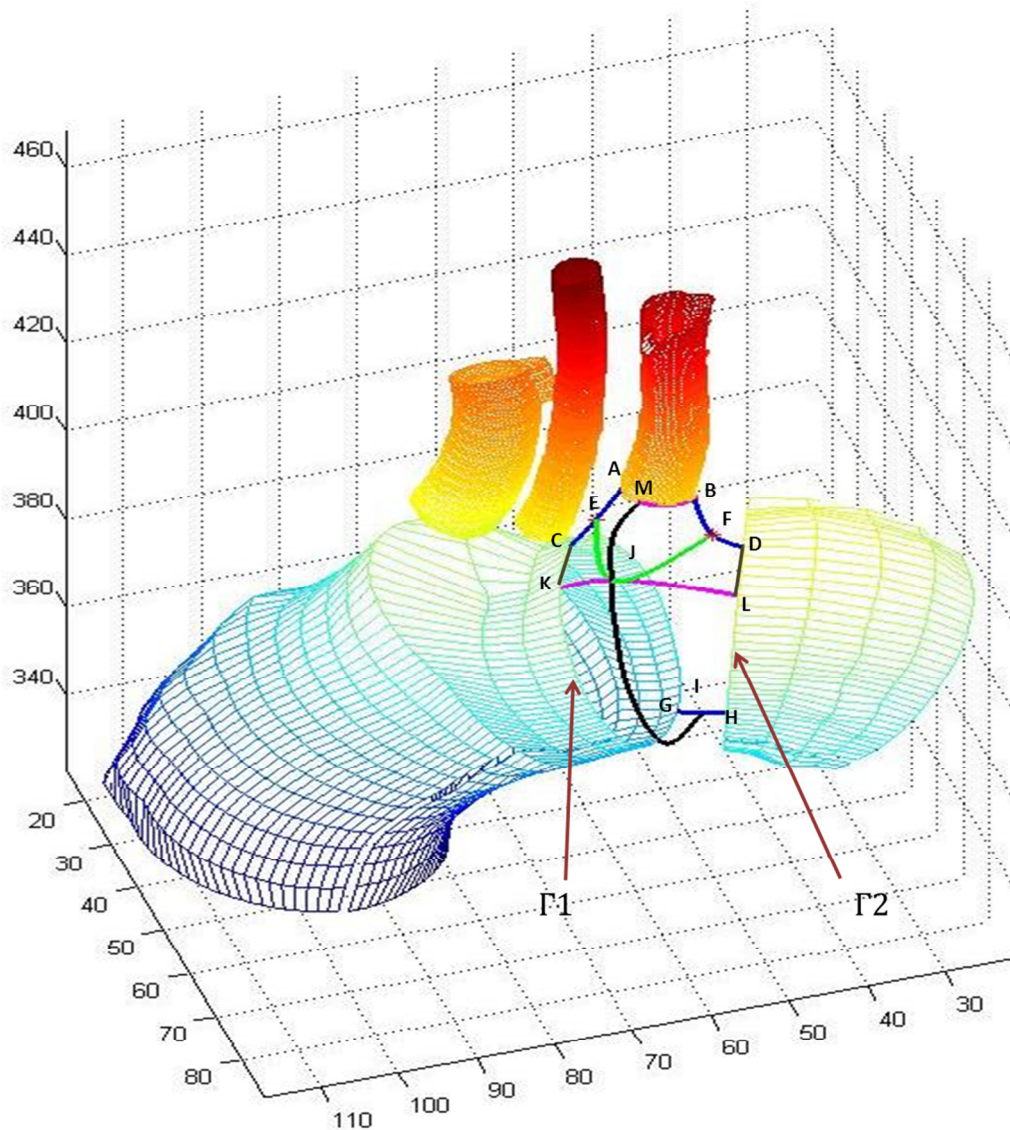


Figure 8.3: Boundary points definition for bilinear patch creation. See main text for details.

The points in cross-sections Γ_1 and Γ_2 that are closest to each other are labelled G and H; point I is defined as the midpoint between G and H.

To determine point J, the information about nodal coordinates obtained from the surface mesh in the transition region is used. Point J is a node on the surface mesh derived in Chapter 5 (Figure 5.30) at minimum distance from the centerline of the

associated branch (subclavian artery). Points K and L are the two nodes at minimum distance from J located on Γ_1 and Γ_2 respectively. Point M is found on the same cross section as A and B, such that it is at minimum distance from J.

The nodes, points and splines described above are shown in Figure 8.4 where the top right and bottom left figures illustrate the front and back views with solid and dashed lines, respectively. The top-right and bottom-left figures show front and back views, respectively.

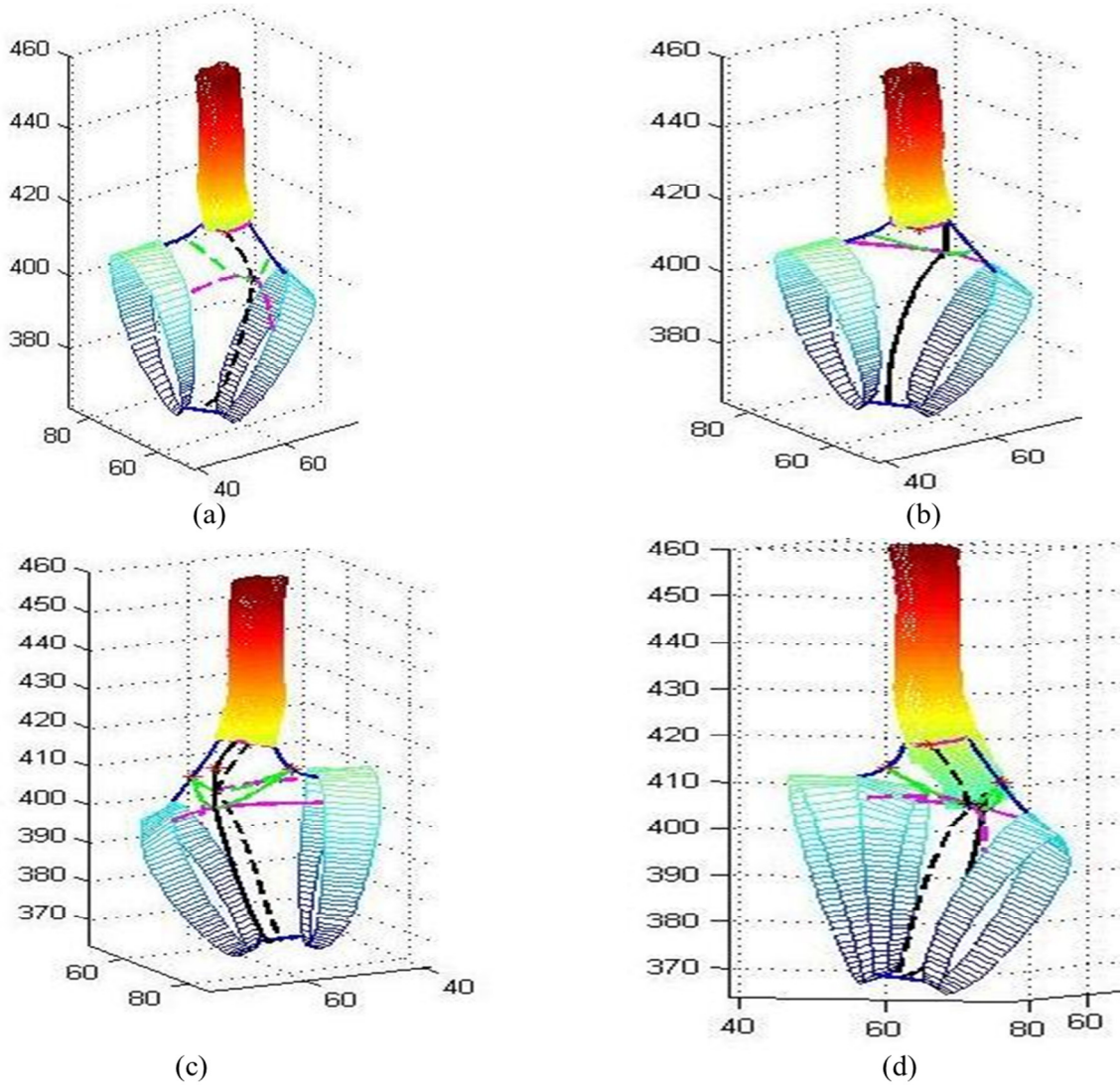


Figure 8.4: Boundary lines drawn for left subclavian artery bifurcation. Units in mm.

Similar constructions are shown Figure 8.5 for the left common coronary artery and the brachiocephalic trunk bifurcations. In this figure, it is assumed that these two branches do not intersect with each other.

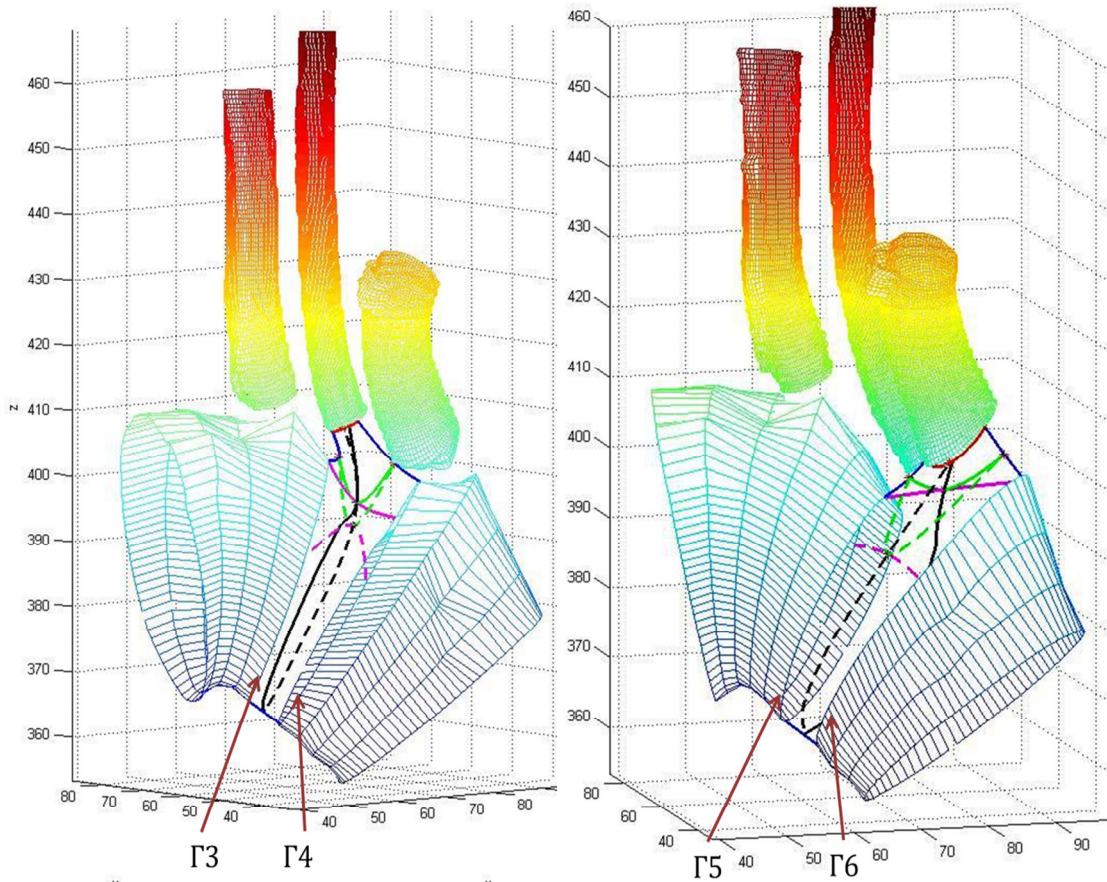


Figure 8.5: Boundary lines drawn for left common coronary artery (left) and brachiocephalic trunk (right). Units in mm.

8.3 Meshing of transition regions

From the boundary lines illustrated in Figure 8.3 to 8.5, 12 sets of four boundary points can readily be defined for each branch. For instance, patch JMBF can be formed in

Figure 8.3. These points can be used towards bilinear patch creation and meshing of the transition regions, as shown in Figure 8.6.

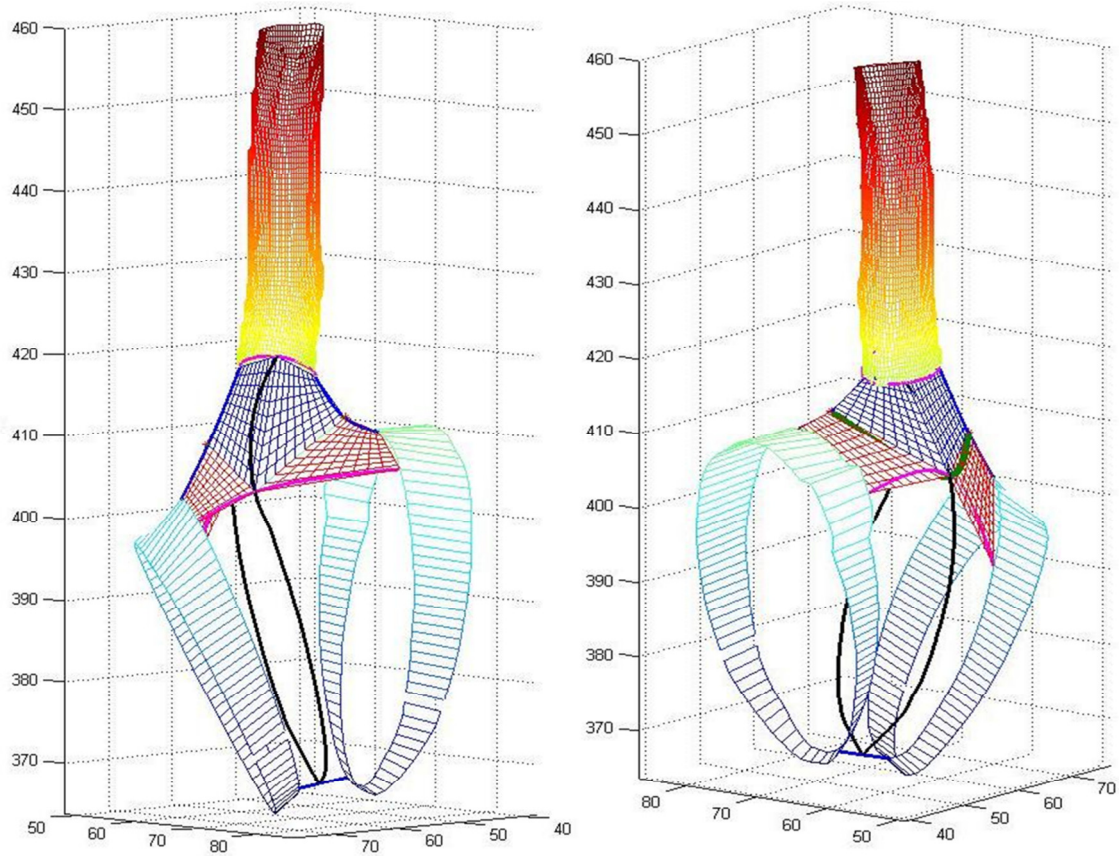


Figure 8.6: Mesh generated using bilinear patch for transition region between aortic arch and left subclavian artery. Left: front view, Right: back view. Units in mm.

Several comments arise from the examination of Figure 8.6. First, patches could be used to cover the entire transition region. Second, discretization of the patches should depend on the discretization of the aorta and of the branches to ensure conforming merging of their respective meshes. Third, use of bilinear patches is obviously suboptimal because the boundary lines of the transition regions are curved, whereas bilinear patches

can only have straight edges. However, bilinear patches were only meant to be used to illustrate the proposed principles of structured mesh merging. Instead of bilinear patches, Coon's patches as described next could be employed. They are expected to provide much improved results once implemented.

8.4 Coon's patches

A Coon's patch is generated by the interpolation of 4 edge curves as shown in Figure 8.7. For any four given curves in space which meet at four corners continuously, a smooth interpolating surface can be generated (Wolf et al., 2003).

The contours are defined as: $\mathbf{C}_1(u)$, $\mathbf{C}_2(u)$, $\mathbf{d}_1(v)$, $\mathbf{d}_2(v)$, with $u, v \in [0,1]$. The surface $\mathbf{X}(u, v)$ interpolating these contours must satisfy the following conditions:

$$\mathbf{X}(u, 0) = \mathbf{C}_1(u), \mathbf{X}(0, v) = \mathbf{d}_1(v), \mathbf{X}(u, 1) = \mathbf{C}_2(u), \mathbf{X}(1, v) = \mathbf{d}_2(v). \quad (8.2)$$

Creation of a linear Coon's patch is done in three steps illustrated in Figure 8.7:

- Generating a temporary surface as:

$$\mathbf{r}_c(u, v) = (1 - v)\mathbf{X}(u, 0) + v\mathbf{X}(u, 1) \quad (8.3)$$

$$\mathbf{r}_d(u, v) = (1 - u)\mathbf{X}(0, v) + u\mathbf{X}(1, v)$$

- Generating an error surface as:

$$\mathbf{r}_{cd}(u, v) = [1 - u \ u] \begin{bmatrix} \mathbf{X}(0,0) & \mathbf{X}(0,1) \\ \mathbf{X}(1,0) & \mathbf{X}(1,1) \end{bmatrix} \begin{bmatrix} 1 - v \\ v \end{bmatrix}$$

Finally, the linear Coon's patch is calculated as the sum of the temporary surfaces minus the error-surface:

$$\mathbf{X}(u, v) = \mathbf{r}_c(u, v) + \mathbf{r}_d(u, v) - \mathbf{r}_{cd}(u, v) \quad (8.4)$$

The derivation of bicubic Coon's patches, using Hermite polynomials, is described in Appendix B. They allow for continuity of tangents between adjacent patches, and therefore are more desirable than linear Coon's patches for the case at hand.

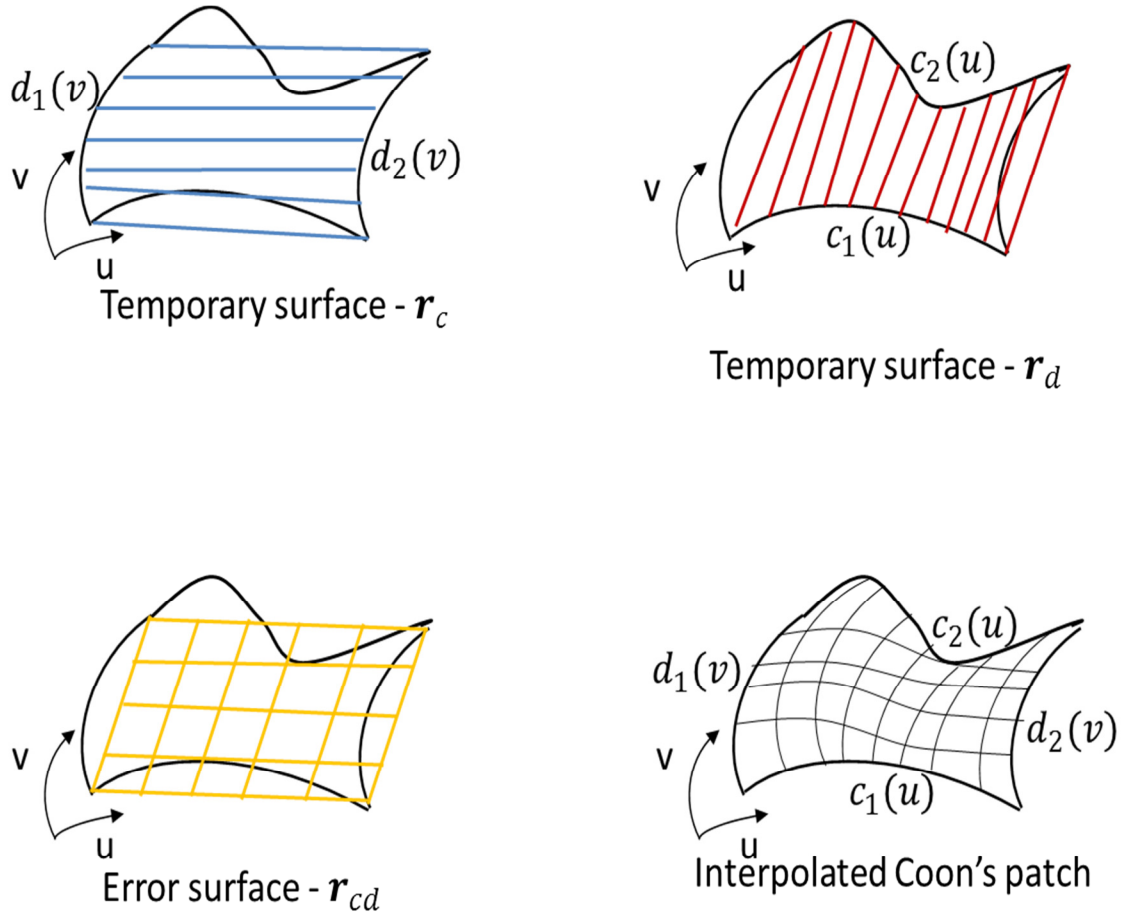


Figure 8.7: Bilinear Coon's patch approach (adapted from Wolf et al., 2003).

9 Discussion

9.1 Significance and novelty of the approach

The proposed methodology effectively streamlines the processing of medical images of the aorta to create structured finite element meshes. It improves on current practice in the following aspects:

- **Reduced number of processing steps:**

Existing software packages, such as commercial Mimics (Materialise HQ, Leuven, Belgium) and ScanIP (Simpleware Ltd., Exeter, UK) or open-source VMTK (Vascular Modelling Toolkit) (Github, 2013) include segmentation and geometric modeling software to create the 3-D geometrical model before generating a mesh. Our approach integrates segmentation and meshing into one effort, resulting in simplified use, and reduced processing time as well as memory usage.

- **Minimal user input:**

Our method is not only fast, it also requires minimal user input. The user needs to select the region of interest, the start and termination points for mesh creation, and define the required mesh density. The default settings for the few parameters that control the process (e.g. regularization parameter and number of iterations for the 3-D GVF calculations; initial mesh radius, distance between consecutive cross-sections, and tortuosity limit) are usually enough to obtain good results. In any case, they can be easily altered as needed through the GUI.

- **No centerline prior information required:**

Recent methods aimed at automating the creation of hexahedral meshes for the aorta (e.g. De Santis, 2011) use the aortic centerline information as a starting point. In our approach, mesh generation and centerline determination are performed simultaneously, making the process efficient and fast.

- **Simplicity of software:**

The whole methodology has been implemented in MATLAB in a seamless GUI that gives the user full and rather intuitive control over the mesh creation.

- **Applicability to multiple image datasets:**

The proposed methodology was successfully applied to different datasets (synthetic, photograph-based, and CT) with various levels of noise and spatial resolutions. As discussed above, the default parameters are robust and may require only minor changes for a wide array of situations.

9.2 Limitations

Beyond apparent limitations pertaining to the fact that not all the ideas presented in this thesis have been fully implemented because of time constraints (see Chapters 7 and 8), the proposed methodology has also more profound limitations. One it that it may be well suited for tubular organs like blood vessels, major airways, the esophagus, the intestines, and perhaps long bones like the femur, but it may not work as well with organs that have more complex shapes like the brain or the liver. Another limitation is that mesh merging for branches may prove time consuming in its current proposed form (Chapter 8). Another limitation previously noted was the difficulty in delineating the intraluminal

thrombus from the actual aortic wall in aneurysms. However, this has more to do with the limitations of the imaging modalities themselves. If the original medical images provided good contrast between both regions, the methodology would provide good segmentation easily.

9.3 Recommendations for future work

The recommended directions for future work are listed below:

9.3.1 Finalize volume meshing capabilities

Direct implementation of the ideas outlined in Chapter 7 will result in full capabilities for refined hexahedral meshing of the aortic wall and the fluid domain.

9.3.2 Develop and finalize branch merging capabilities

The ideas proposed in Chapter 8 need to be implemented, thoroughly tested and generalized to handle volume meshing. Challenges will arise from the difficulty to reliably image or predict the blood vessel thickness in the transition regions around branches.

9.3.3 Investigate segmentation tools other than the 3-D GVF

Although the 3-D GVF field combined with structured mesh creation has proved to be a powerful segmentation tool, it is sensitive to the noise present in the medical images. To combat the influence of this noise, it may be valuable to explore the performance of the proposed ray-tracing method when using enhanced versions of the GVF, such as

provided by the vector field convolution (VFC) and the generalized gradient vector field (GGVF) methods (Li & Acton, 2007; Prabhu Britto, 2005).

9.3.4 Considering other aortic datasets and other organs

The number of aortic datasets used in this work was small and more cases should be processed to further establish the robustness of the methods proposed. This should be done not only with CT, but with MR datasets as well, since MR imaging is also frequently used. Other organs with tube-like geometries are prime targets for expanding the range of applications of the proposed methodology. More complex geometries may possibly be investigated too, given the potential savings in processing time compared to existing methods.

9.3.5 Toward a complete mesh generation package

After completion of the above recommended work, the opportunity may arise to put together a “Structured bio-mesh generation package” that could be shared with the research community and/or commercialized. At minimum, the current contribution may be added to the VMTK project (GitHub, 2013).

10 Conclusion

The techniques presented in this thesis represent a step toward direct structured finite element mesh generation from patient-specific 3-D medical images of the aorta. The properties of the 3-D gradient vector flow field lie at the foundation of the proposed methodology, although other types of fields might possibly be used in the future. Rather naïve attempts at using this field as a force or displacement field to deform an initial finite element mesh failed, but paved the way to a novel heuristic, perhaps, but very efficient, incremental mesh construction methodology. A major challenge during the creative process was to maintain control over the tortuosity of the reconstructed centerline of the aorta. This was finally achieved with the proposed orientation control algorithm. Another related challenge was the possible folding of the final mesh. This was mediated by down-sampling the number of points on the aortic centerline and by longitudinal interpolation between cross-sections of the mesh. It is also worth noting that building a GUI around the whole process made it even more effective and seamless. Although, as outlined above, some more work will be needed to finalize the implementation of the methodology toward hexahedral meshing of branched blood vessels, the quality of the underlying surface meshes was found to be very good in the datasets considered.

Faster, more effective and easier to use meshing tools will eventually help to bring engineering modeling and simulation into the cardiovascular clinical arena as diagnosis and surgical planning supports. It is the ultimate goal of the present work to contribute, albeit in a small way, to better disease management.

Appendix A: 3-D GVF computation

As shown in Chapter 3, determining the GVF for a function f in 3-D amounts to solving the following coupled Euler equations in u , v and w :

$$\begin{cases} \|\nabla f\|^2 \left(u - \frac{\partial f}{\partial x} \right) - \mu \nabla^2 u = 0 \\ \|\nabla f\|^2 \left(v - \frac{\partial f}{\partial y} \right) - \mu \nabla^2 v = 0 \\ \|\nabla f\|^2 \left(w - \frac{\partial f}{\partial z} \right) - \mu \nabla^2 w = 0 \end{cases}$$

The solution of these equations is simplified by redefining u , v and w as functions of space and time, and by subtracting their time derivatives from the corresponding equations, such that:

$$\begin{cases} -\frac{\partial u}{\partial t} + \|\nabla f\|^2 \left(u - \frac{\partial f}{\partial x} \right) - \mu \nabla^2 u = 0 \\ -\frac{\partial v}{\partial t} + \|\nabla f\|^2 \left(v - \frac{\partial f}{\partial y} \right) - \mu \nabla^2 v = 0 \\ -\frac{\partial w}{\partial t} + \|\nabla f\|^2 \left(w - \frac{\partial f}{\partial z} \right) - \mu \nabla^2 w = 0 \end{cases}$$

Therefore, finding the 3-D GVF amounts to solving the following equations:

$$\begin{cases} \frac{\partial u}{\partial t} = \|\nabla f\|^2 \left(u - \frac{\partial f}{\partial x} \right) - \mu \nabla^2 u \\ \frac{\partial v}{\partial t} = \|\nabla f\|^2 \left(v - \frac{\partial f}{\partial y} \right) - \mu \nabla^2 v \\ \frac{\partial w}{\partial t} = \|\nabla f\|^2 \left(w - \frac{\partial f}{\partial z} \right) - \mu \nabla^2 w \end{cases}$$

Using finite differences to approximate the previous equations, with indices i, j, k and n corresponding to x, y, z and t with intervals $\Delta x, \Delta y, \Delta z$ and Δt , respectively, one obtains:

$$\frac{\partial u}{\partial t} = \frac{1}{\Delta t} (u_{i,j,k}^{n+1} - u_{i,j,k}^n)$$

$$\frac{\partial v}{\partial t} = \frac{1}{\Delta t} (v_{i,j,k}^{n+1} - v_{i,j,k}^n)$$

$$\frac{\partial w}{\partial t} = \frac{1}{\Delta t} (w_{i,j,k}^{n+1} - w_{i,j,k}^n)$$

$$\nabla^2 u = \frac{1}{\Delta x \Delta y \Delta z} (u_{i+1,j,k}^n + u_{i,j+1,k}^n + u_{i,j,k+1}^n + u_{i-1,j,k}^n + u_{i,j-1,k}^n + u_{i,j,k-1}^n - 6u_{i,j,k}^n)$$

$$\nabla^2 v = \frac{1}{\Delta x \Delta y \Delta z} (v_{i+1,j,k}^n + v_{i,j+1,k}^n + v_{i,j,k+1}^n + v_{i-1,j,k}^n + v_{i,j-1,k}^n + v_{i,j,k-1}^n - 6v_{i,j,k}^n)$$

$$\nabla^2 w = \frac{1}{\Delta x \Delta y \Delta z} (w_{i+1,j,k}^n + w_{i,j+1,k}^n + w_{i,j,k+1}^n + w_{i-1,j,k}^n + w_{i,j-1,k}^n + w_{i,j,k-1}^n - 6w_{i,j,k}^n)$$

Since function f is known in space and fixed in time, it is assumed that its derivatives can easily be obtained, allowing us to set:

$$b_{i,j,k} = b(x, y, z) = \frac{\partial f^2}{\partial x} + \frac{\partial f^2}{\partial y} + \frac{\partial f^2}{\partial z}$$

along with

$$c_{i,j,k}^u = b(x, y, z) \frac{\partial f}{\partial x},$$

$$c_{i,j,k}^v = b(x, y, z) \frac{\partial f}{\partial y},$$

$$c_{i,j,k}^w = b(x, y, z) \frac{\partial f}{\partial z}.$$

Finally, one obtains iterative approximations of u , v and w at time $n + 1$ from those at time n as follows:

$$\begin{aligned} u_{i,j,k}^{n+1} = & (1 - b_{i,j,k}\Delta t)u_{i,j,k}^n \\ & + r(u_{i+1,j,k}^n + u_{i,j+1,k}^n + u_{i,j,k+1}^n + u_{i-1,j,k}^n + u_{i,j-1,k}^n + u_{i,j,k-1}^n - 6u_{i,j,k}^n) \\ & + c_{i,j,k}^u \Delta t \end{aligned}$$

$$\begin{aligned} v_{i,j,k}^{n+1} = & (1 - b_{i,j,k}\Delta t)v_{i,j,k}^n \\ & + r(v_{i+1,j,k}^n + v_{i,j+1,k}^n + v_{i,j,k+1}^n + v_{i-1,j,k}^n + v_{i,j-1,k}^n + v_{i,j,k-1}^n - 6v_{i,j,k}^n) \\ & + c_{i,j,k}^v \Delta t \end{aligned}$$

$$\begin{aligned} w_{i,j,k}^{n+1} = & (1 - b_{i,j,k}\Delta t)w_{i,j,k}^n \\ & + r(w_{i+1,j,k}^n + w_{i,j+1,k}^n + w_{i,j,k+1}^n + w_{i-1,j,k}^n + w_{i,j-1,k}^n + w_{i,j,k-1}^n - 6w_{i,j,k}^n) \\ & + c_{i,j,k}^w \Delta t \end{aligned}$$

$$\text{with } r = \frac{\mu\Delta t}{\Delta x\Delta y\Delta z}.$$

The equations above were programmed in MATLAB to obtain the 3-D GVF from 3-D images. Solutions at time 0 were initialized from the values of the gradient of f , and the notion of time was replaced by the notion of iterations.

Appendix B: Bicubic Coon's patch

Contiguous, linearly interpolated Coon's patches only share straight edges, and first derivatives are discontinuous between adjacent Coons patches. As explained in (Wolf et al., 2003), a solution to this problem consists in using cubic Hermite interpolation, such that: $\mathbf{H}_i(t)$, $i = 0, \dots, 3$ with

$$H_0(t) = 2t^3 - 3t^2 + 1$$

$$H_1(t) = t^3 - 2t^2 + 1$$

$$H_2(t) = t^3 - t^2$$

$$H_3(t) = -2t^3 + 3t^2 + 1$$

The first partial derivatives $X_n(u, v)$ with respect to $n \in \{u, v\}$ perpendicular to the patch boundary are interpolated from the tangents at their endpoints ($u, v \in \{0, 1\}$) as:

$$\mathbf{X}_u(0, v) = \mathbf{X}_u(0, 0)H_0(v) + \mathbf{X}_u(0, 1)H_3(v)$$

$$\mathbf{X}_u(1, v) = \mathbf{X}_u(1, 0)H_0(v) + \mathbf{X}_u(1, 1)H_3(v)$$

$$\mathbf{X}_v(u, 0) = \mathbf{X}_v(0, 0)H_0(u) + \mathbf{X}_v(1, 0)H_3(u)$$

$$\mathbf{X}_v(u, 1) = \mathbf{X}_v(0, 1)H_0(u) + \mathbf{X}_v(1, 1)H_3(u)$$

As in the linear case, the two temporary surfaces $\mathbf{h}_c(u, v)$ and $\mathbf{h}_d(u, v)$ are calculated as:

$$\mathbf{h}_c(u, v) = \mathbf{X}(0, v)H_0(u) + \mathbf{X}_u(0, v)H_1(u) + \mathbf{X}_u(1, v)H_2(u) + \mathbf{X}(1, v)H_3(u)$$

$$\mathbf{h}_d(u, v) = \mathbf{X}(u, 0)H_0(v) + \mathbf{X}_u(u, 0)H_1(v) + \mathbf{X}_v(u, 1)H_2(v) + \mathbf{X}(u, 1)H_3(v)$$

The error surface is calculated as:

$$\begin{aligned} & \mathbf{h}_{cd}(u, v) \\ &= [H_0(u) \quad H_1(u) \quad H_2(u) \quad H_3(u)] \begin{bmatrix} \mathbf{X}(0,0) & \mathbf{X}_v(0,0) & \mathbf{X}_v(0,1) & \mathbf{X}(0,1) \\ \mathbf{X}_u(0,0) & 0 & 0 & \mathbf{X}_u(0,1) \\ \mathbf{X}_u(1,0) & 0 & 0 & \mathbf{X}_u(1,1) \\ \mathbf{X}(1,0) & \mathbf{X}_v(1,0) & \mathbf{X}_v(1,1) & \mathbf{X}(1,1) \end{bmatrix} \begin{bmatrix} H_0(v) \\ H_1(v) \\ H_2(v) \\ H_3(v) \end{bmatrix} \end{aligned}$$

Finally, the bicubic Coon's patch is combined as (see Figure 8.7):

$$\mathbf{X}(u, v) = \mathbf{h}_c(u, v) + \mathbf{h}_d(u, v) - \mathbf{h}_{cd}(u, v).$$

Bibliography

- Antiga, L., Piccinelli, M., Botti, L., Ene-Iordache, B., Remuzzi, A., Steinman, D., 2008. An image-based modeling framework for patient-specific computational hemodynamics. *Medical & biological engineering & computing*, 46(11), pp.1097–1112.
- Antiga, L., Ene-Iordache, B., Caverni, L., Cornalba, G. P., Remuzzi, A., 2002. Geometric reconstruction for computational mesh generation of arterial bifurcations from CT angiography. *Computerized medical imaging and graphics: the official journal of the Computerized Medical Imaging Society*, 26(4), pp.227–235.
- Antiga, L. & Steinman, D. a, 2004. Robust and objective decomposition and mapping of bifurcating vessels. *IEEE transactions on medical imaging*, 23(6), pp.704–713.
- Appel, A., 1968. Some techniques for shading machine renderings of solids. In *AFIPS 1968 Spring Joint Computer Conf.* pp. 37–45.
- Auer, M. & Gasser, T.C., 2010. Reconstruction and finite element mesh generation of abdominal aortic aneurysms from computerized tomography angiography data with minimal user interactions. *IEEE transactions on medical imaging*, 29(4), pp.1022–1028.
- Ayyalasomayajula, A., Polk, A., Basudhar, A., Missoum, S., Nissim, L., Vande Geest, J. P., 2010. Three dimensional active contours for the reconstruction of abdominal aortic aneurysms. *Annals of biomedical engineering*, 38(1), pp.164–176.
- Azar, F.S., Metaxas, D.N. & Schnall, M.D., 2002. Methods for modeling and predicting mechanical deformations of the breast under external perturbations. *Medical image analysis*, 6(1), pp.1–27.

- Baghdadi, L., Steinman, D. a & Ladak, H.M., 2005. Template-based finite-element mesh generation from medical images. *Computer methods and programs in biomedicine*, 77(1), pp.11–21.
- Bakker, A., 2008. Computational fluid dynamic lectures. Available at: <http://www.bakker.org/dartmouth06/engs150/>, [Accessed November 20, 2013].
- Barber, D.C. & Hose, D.R., 2005. Automatic segmentation of medical images using image registration: diagnostic and simulation applications. *Journal of medical engineering & technology*, 29(2), pp.53–63.
- Bardinet, E., Cohen, L. & Ayache, N., 1995. A parametric deformable model to fit unstructured 3d data. *NRIA, Tech. Rep*, I. 2617.
- Battle, X., Cunningham, G. & Hanson, K., 1997. 3D tomographic reconstruction using geometrical models. *Proc SPIE*, 3034, pp.346–357.
- Benedict Verheghe, 2010. pyformex. Available at: <http://www.nongnu.org/pyformex/> [Accessed September 27, 2013].
- Bhatti, M.A., 2006. *Fundamental Finite Element Analysis and Applications with Mathematica and MATLAB computations*, Wiley.
- De Biasi, H., von Wangenheim, A., Silveira, P.G., Comunello, E., 2002. 3D reconstruction of abdominal aortic aneurysms. In *Proceedings of 15th IEEE Symposium on Computer-Based Medical Systems (CBMS)*. IEEE Computer. Soc, pp. 245–250.
- Biesdorf, A., Rohr K, Feng D, Tengg-Kobligk H, Rengier F, Böckler D, Kauczor HU, Wörz S., 2012. Segmentation and quantification of the aortic arch using joint 3D model-based segmentation and elastic image registration. *Medical image analysis*, 16(6), pp.1187–201.

- Borgefors, G., 1984. Distance transformations in arbitrary dimensions. *Compute. Vision, Graphics, and Image Processing*, 27, pp.321–345.
- Boutchko, R., Sitek, a & Gullberg, G.T., 2013. Practical implementation of tetrahedral mesh reconstruction in emission tomography. *Physics in medicine and biology*, 58(9), pp.3001–22.
- Bowden, A., Rabbitt, R. & Weiss, J., 1998. Anatomical registration and segmentation by warping template finite element models. *Proc Laser-Tissue Interact*, IX, pp.469 – 476.
- Bro-Nielsen, M. & Cotin, S., 1996. Real-time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation. *Computer Graphics Forum*, 15, pp.57–66.
- Brown, L., 1992. A survey of image registration techniques. *ACM Comput Surv (CSUR)*, 24(4), pp.325 – 376.
- De Bruijne, M., van Ginneken, B., Niessen, W. J., Maintz, JB. A., Viergever, M.A., 2002. Active shape model based segmentation of abdominal aortic aneurysms in CTA images, 4684, pp.463–474.
- Canny, J., A., 1986. Computational Approach To Edge Detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6), pp.679–698.
- Chae, S.W. & Kwon, G.W., 2003. Finite element modeling from medical images for biomechanical analysis. *Computational Fluid and Solid Mecha*, pp.1638–1641.
- Chan, T.F. & Vese, L. A, 2001. Active contours without edges. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 10(2), pp.266–277.

- Chunming, Li, 2005. Level Set Evolution without Re-Initialization: A New Variational Formulation. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, 1, pp.430–436.
- Cline, H., Lorensen W.E, Souza S.P, Jolesz F.A, Kikinis R., Gerig G., Kennedy T.E., 1991. 3-D surface rendered MR images of the brain and its vasculature. *JCAT*, 15, pp.344–351.
- Cohen, L.D., 1991. On active contour models and balloons. *Computer Vision, Graphics and Image Processing*, 53(2), pp.211–218.
- Cohen, L.D. & Cohen, I., 1993. Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11).
- Courant, R. & Hilbert, D., 1953. *Methods of Mathematical Physics*, New York.
- Couteau, B., Payan, Y. & Lavallee, S., 2000. The mesh-matching algorithm: an automated 3-D mesh generator for finite element structures. *Biomechanics*, 33(8), pp.1005–1009.
- Depa, M., Sabuncu, Mert R., Holmvang, G., Nezafat, R., Schmidt, E. J., Golland, P., 2010. Robust Atlas-Based Segmentation of Highly Variable Anatomy : Left Atrium Segmentation. In *Statistical Atlases and Computational Models of the Heart Lecture Notes in Computer Science*. Berlin Heidelberg: Springer-Verlag, pp. 85–94.
- Dhole, N.E., 2010. Developmen and validation of a finite element model of a transprot aircraft seat undert part 25.562 dynamic test conditions. Wichita State University.
- Ferrant, M., Nabavi, A., Macq, B., Jolesz, F.A., Kikinis, R., Warfield, S. K., 2001. Registration of 3-D intraoperative MR images of the brain using a finite-element biomechanical model. *IEEE transactions on medical imaging*, 20(12), pp.1384–97.

- Fisher, R., Perkins, S., Walker, A., Wolfart, E., 2003. Canny Edge Detector. Available at: <http://homepages.inf.ed.ac.uk/rbf/HIPR2/canny.htm>.
- Foo, T.K., Ho, V.B. & Choyke, P.L., 1999. Contrast-enhanced carotid MR angiography. *Imaging principles and physics*, 9, pp.263–284.
- Forsyth, J., Borsic, A., Halter, R J., Hartov, A., Paulsen, K. D., 2011. Optical breast shape capture and finite-element mesh generation for electrical impedance tomography. *Physiological measurement*, 32(7), pp.797–809.
- Frangi, A.F., Niessen, W. J., Vincken, K. L., Viergever, M. A., 1998. Multiscale vessel enhancement filtering 1 Introduction 2 Method. , 1496, pp.130–137.
- Fujioka, H., Oka, K. & Tanishita, K., 2000. Clinical application of computational fluid and solid mechanics for the cardiovascular system, Tokyo: Springer-Verlag.
- Gaens, T., Maes, F., Vandermeulen, D., Suetens, P., 1998. Non-rigid multimodal image registration using mutual information., *Proc. First International Conference MICCAI*, pp.1099 – 1106.
- Galante, V. , Corsi, C., Veronesi, F., Russo, V., Fattori, R., Lamberti, C., 2007. Dynamic Characterization of Aorta Morphology and Function in Presence of an Aneurysm DEIS , University of Bologna , Bologna , Italy. , pp.765–768.
- Gallier, J., 2006. Mesh generation from imaging data, University of Pennsylvania.
- Gao, F., Watanabe, M. & Matsuzawa, T., 2005. Three-dimensional computational mechanical analysis for 3-layered aortic arch model under steady and unsteady flow with fluid-structure interactions. *Eighth International Conference on High-Performance Computing in Asia-Pacific Region (HPCASIA'05)*, pp.–161-169.
- Gao, F, Watanabe, M, Matsuzawa, T., 2006. Stress analysis in a layered aortic arch model under pulsatile blood flow. *BioMedical Engineering OnLine*, pp.5–25.

- GitHub, 2013. vmtk. Available at: <http://www.vmtk.org/Main/Overview/> [Accessed September 27, 2013].
- Goldwin, J.D., 1990. Conventional CT of the aorta. *Journal of Thoracic Imaging*, 5(4), pp.18–31.
- Gonzales, M.J., Gregory S., Adarsh K., Johan H., René J., Paul S., Wouter-Jan Rappele, f, et al., 2013. A three-dimensional finite element model of human atrial anatomy: new methods for cubic Hermite meshes with extraordinary vertices. *Medical image analysis*, 17(5), pp.525–37.
- Grosland, N.M., Bafna, R. & Magnotta, V. a., 2009. Automated hexahedral meshing of anatomic structures using deformable registration. *Computer Methods in Biomechanics and Biomedical Engineering*, 12(1), pp.35–43.
- Hackam, D.G., Thiruchelvam, D. & Redelmeier, D. a, 2006. Angiotensin-converting enzyme inhibitors and aortic rupture: a population-based case-control study. *Lancet*, 368(9536), pp.659–65.
- Hager A., Kaemmerer H., Rapp-Bernhardt U., Blücher S., Rapp K., Bernhardt T.M., Galanski M, Hess J., 2002. Diameters of the thoracic aorta throughout life as measured with helical computed tomography. *The Journal of Thoracic and Cardiovascular Surgery*, 123(6), pp.1060–1066.
- Hansen, G.A., Douglass, R.W. & Zardecki, A., 2005. Mesh Enhancement: Selected Elliptic Methods, Foundations And Applications, Imperial College Press.
- Helderman F, Manoch I.J, Breeuwer M., Kose U., Boersma H., van Sambeek M.R, Pattynama P.M, Schouten O., et al., 2010. Predicting patient-specific expansion of abdominal aortic aneurysms. *European journal of vascular and endovascular surgery : the official journal of the European Society for Vascular Surgery*, 40(1), pp.47–53..
- Isgum, I., Staring, M., Rutten, A., Prokop, M., Viergever, M.A., van Ginneken, B., 2009. Multi-atlas-based segmentation with local decision fusion--application to cardiac and

aortic segmentation in CT scans. *IEEE transactions on medical imaging*, 28(7), pp.1000–1010.

Jodoin, P.-M. , Alain, L., Yvon, V., Olivier, B., Steinmetz, E., 2008. Markovian method for 2D, 3D and 4D segmentation of MRI, *15th IEEE International Conference on Image Processing, ICIP*, pp.3012–3015.

Kass, M., Witkin, A. & Terzopoulos, D., 1988. Snakes : Active Contour Models. *International Journal of Computer Vision*, 331, pp.321–331.

Katz, T. & Merickel, M.B., 1990. Translation-invariant aorta segmentation from magnetic resonance images. *International Joint Conference on Neural Networks*, 1, pp.327–333.

Kim, H.C., Seol, Y. H., Choi, S. Y., Oh, J. S., Kim, M. G., Sun, K., 2007. A study of AAA image segmentation technique using geometric active contour model with morphological gradient edge function. *Conference proceedings : Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 2007, pp.4437–4440.

Kirbas, C. & Quek, F.K.H., 2003. Vessel extraction techniques and algorithms: a survey. *Third IEEE Symposium on Bioinformatics and Bioengineering, 2003. Proceedings.*, pp.238–245.

Klein, A., Renema W.K., Oostveen L.J., Schultze Koo L.J., Slump C.H., Hu X.P., Clough A.V. , 2008. A segmentation method for stentgrafts in the abdominal aorta from ECG-gated CTA data, *Proc. SPIE 6916, Medical Imaging 2008: Physiology, Function, and Structure from Medical Images*.

Knupp, P.M., 2007. Remarks on Mesh Quality. In *Aerospace Sciences Meeting and Exhibit*. pp. 7–10.

- Kovacevic, A., 2005. Boundary adaptation in grid generation for CFD analysis of screw compressors. *International Journal for Numerical Methods in Engineering*, 64(3), pp.401–426.
- Kovacs, T., 2010. *Automatic Segmentation of the Vessel Lumen from 3D CTA Images of Aortic Dissection presented by*. Budapest University of Technology and Economics.
- Kurugol, S., San Jose E. R., Ross, J., Washko, G. R., 2012. Aorta segmentation with a 3D level set approach and quantification of aortic calcifications in non-contrast chest CT. *Conference proceedings: .Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 2012, pp.2343–2346.
- Kyriacou, S.K., Davatzikos, C., Zinreich, S. J., Bryan, R. N., 1999. Nonlinear elastic registration of brain images with tumor pathology using a biomechanical model. *IEEE transactions on medical imaging*, 18(7), pp.580–592.
- Li B. & Acton S. T., 2006. Vector field convolution for image segmentation using snakes. *Proc. IEEE Int. Conf. Image Processing (CIP)*, 16(8), pp 1637-1640.
- Li, B. & Acton, S.T., 2007. Active contour external force using vector field convolution for image segmentation. *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, 16(8), pp.2096–2106.
- Loncaric, S., Subasic, M. & Sorantin, E., 2000. 3-D deformable model for aortic aneurysm segmentation from CT images. *Proceedings of the 22nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1, pp.398–401.
- Longest, P.W. & Vinchurkar, S., 2007. Effects of mesh style and grid convergence on particle deposition in bifurcating airway models with comparisons to experimental data. *Medical engineering & physics*, 29(3), pp.350–566.

- Lu, Y., Bo. Y., Levon, V., Andrzej K., Yuesheng, X., Xiaofei, Hu, David, F., 2009. Tomographic mesh generation for OSEM reconstruction of SPECT images, *Proc. SPIE 7258, Medical Imaging : Physics of Medical Imaging*, 7258, pp.725854–725857.
- M. Kass, A. Witkin, & Terzopoulos, D., 1988. Snakes: Active contour models, *International Journal of Computer Vision*, 1(4), pp.321– 331.
- Macía, I. et al., 2009. Segmentation of Abdominal Aortic Aneurysms in CT Images using a Radial Model Approach. *IDEAL'09 Proceedings of the 10th international conference on Intelligent data engineering and automated*, pp.664–671.
- Magee, D., Bulpitt, A. & Berry, E., 2001. Combining 3D Deformable Models and Level Set Methods for the Segmentation of Abdominal Aortic Aneurysms *Proceedings of the British Machine Vision Conference, BMVC*. pp. 333–341.
- Mahmoudi, H., 2013. *Modeling of Bonnet in LS-DYNA for Pedestrian Research*. Chalmers.
- Malladi, R., 1995. Shape modeling with front propagation: a level set approach. *IEEE Transactions on pattern and machine intelligence*, 17(2), pp.158–175.
- Mansi, T., Di Martino, E. S., Amon, C. H., Muluk, S. C., Finol, E.A., 2012. An integrated framework for finite-element modeling of mitral valve biomechanics from medical images: application to MitralClip intervention planning. *Medical image analysis*, 16(7), pp.1330–46.
- Marshall, D., 2008. Region Growing. Available at:
http://www.cs.cf.ac.uk/Dave/Vision_lecture/node35.html.
- Martufi, G. , Di Martino, E. S., Amon, C.H., Muluk, S. C., Finol, E. A. 2009. Three-dimensional geometrical characterization of abdominal aortic aneurysms: image-based wall thickness distribution. *Journal of biomechanical engineering*, 131(6), p.061015.

- McDonald, D., 1974. Blood flow in arteries. *Baltimore, MD: Williams & Wilkins*, pp.258–259.
- McInerney, T. & Terzopoulos, D., 1996. Deformable models in medical image analysis: a survey. *Medical image analysis*, 1(2), pp.91–108.
- Metcalf, D., Holt, P.J.E. & Thompson, M.M., 2011. The management of abdominal aortic aneurysms. *British Medical Journal (BMJ)*, 342(d:1384), pp.644–649.
- Meyer, M., Kirby, R.M. & Whitaker, R., 2007. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE transactions on visualization and computer graphics*, 13(6), pp.1704–11.
- Mohamed, A. & Davatzikos, C., 2004. Finite element mesh generation and remeshing from segmentation. *IEEE Biomedical imaging*, pp.420–423.
- Moll, F.L., Powell J.T., Fraedrich G., Verzini F., Haulon S., Waltham M., van Herwaarden J.A, Holt P.J, et al., 2011. Management of abdominal aortic aneurysms clinical practice guidelines of the European society for vascular surgery. *European journal of vascular and endovascular surgery : the official journal of the European Society for Vascular Surgery*, 41 Suppl 1, pp.S1–S58.
- Natarajan, A., Magnotta, V. a. & Grosland, N.M., 2010. Hexahedral Meshing of Subject-Specific Anatomic Structures Using Registered Building Blocks. *ASME 2010 Summer Bioengineering Conference, Parts A and B*, pp.457–458. A
- Olabarriaga, D., Breeuwer, M. & Niessen, W.J., 2004. Segmentation of Abdominal Aortic Aneurysms with a Non-parametric Appearance Model, *Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis, Lecture Notes in Computer Science*, 3117, pp. 257-268
- Osher, S.J.A.S., 1988. Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, 79, pp.12–49.

- OsiriX Team, 2012. DICOM sample image sets. Available at: <http://www.osirix-viewer.com/datasets/>, [Accessed November 30, 2012].
- Papademetris, X., Sinusas, A.J. Dione, D. P., Constable, R T., Duncan, J. S, 2002. Estimation of 3-D left ventricular deformation from medical images using biomechanical models. *IEEE transactions on medical imaging*, 21(7), pp.786–800.
- Park, J., Young, A., Axel, L., Metaxas, D., 1996. Deformable Models with Parameter Functions for Cardiac Motion Analysis from Tagged MRI Data Deformable Models with Parameter Functions for Cardiac Motion Analysis from Tagged MRI Data. *IEEE Transactions on Medical Imaging*, 15(3), pp.278–289.
- Peiró, J., Sherwin, S.J. & Giordana, S., 2008. Automatic reconstruction of a patient-specific high-order surface representation and its application to mesh generation for CFD calculations. *Medical & biological engineering & computing*, 46(11), pp.1069–83.
- Pham, D.L., Xu, C. & L., Prince, J., 2000. Current methods in medical image segmentation. *Annu. Rev. Biomed. Eng.*, 02, pp.315–337.
- Pohle, R. & Toennies, K.D., 2001. Segmentation of medical images using adaptive region growing, Proc. *SPIE* 4322, Medical Imaging , Image processing 1337 , pp.1337–1346.
- Prabhu Britto, A. & Ravindran G., 2005. Chromosome Segmentation and Investigations using Generalized Gradient Vector Flow Active Contours. *Online Journal of Health & Allied Sciences*, 4(2), ISSN 0972-5997.
- Putter, S. De., Vosse, F. N. Van De., Gerritsen, F. A., Laffargue, F., Breeuwer, M., 2006. Computational Mesh Generation for Vascular Structures with Deformable Surfaces. *International Journal of Computer Assisted Radiology and Surgery*, 1(1), pp.39–49.

- Radaelli, A.G. & Peiro, J., 2010. On the segmentation of vascular geometries from medical images. *International Journal of Numerical Biomedical Engineering*, pp.26–34.
- Rafael E. Gonzalez and Richard, 1993. *Digital Image Processing*, Addison Wesley.
- Ramos, A & Simões, J. A, 2006. Tetrahedral versus hexahedral finite elements in numerical modelling of the proximal femur. *Medical engineering & physics*, 28(9), pp.916–924.
- Ramsey, S.D., Potter, K. & Hansen, C., 2012. Ray Bilinear Patch Intersections. *Journal of Graphics Tools*, 9(3), pp.41–47.
- Ricotta, J.J. et al., 2008. Cardiovascular disease management: the need for better diagnostics. *Medical & biological engineering & computing*, 46(11), pp.1059–68.
- Rueckert, D., Burger, P., Forbat, S. M., Mohiaddin, R. D., Yang, G. Z. , 1997. Automatic tracking of the aorta in cardiovascular MR images using deformable models. *IEEE transactions on medical imaging*, 16(5), pp.581–90.
- De Santis, G., De Beule, M., Van Canneyt, K., Segers, P., Verdonck, P., Verhegghe, B., 2011. Full-hexahedral structured meshing for image-based computational vascular modeling. *Medical engineering & physics*, 33(10), pp.1318–1325.
- De Santis, G., De Beule, M., Segers, P., Verdonck, P., Verhegghe, B., 2010. Patient-specific computational fluid dynamics: structured mesh generation from coronary angiography. *Medical & biological engineering & computing*, 48(4), pp.371–380.
- De Santis, G., De Beule, M., Segers, P., Verdonck, P., Verhegghe, B., 2011. Patient-specific computational haemodynamics: generation of structured and conformal hexahedral meshes from triangulated surfaces of vascular bifurcations. *Computer methods in biomechanics and biomedical engineering*, 14(9), pp.797–802.

- Sarry, L. & Boire, J.Y., 2001. Three-dimensional tracking of coronary arteries from biplane angiographic sequences using parametrically deformable models. *IEEE transactions on medical imaging*, 20(12), pp.1341–1351.
- Shepherd, J.F., Zhang, Y., Tuttle, C. J., Silva, C. T., 2007. Quality Improvement and Boolean-Like Operations in Hexahedral Meshes. In *10th ISGG conference on numerical grid generation*, Forth. Crete, Greece, pp. 16–20.
- Shim, M.-B., Gunay, M. & Shimada, K., 2009. Three-dimensional shape reconstruction of abdominal aortic aneurysm. *Computer-Aided Design*, 41(8), pp.555–565.
- Shivanna, K.H., Tadepalli, S.C. & Grosland, N.M., 2010. Feature-Based Multiblock Finite Element Mesh Generation. *Computer aided design*, 42(12), pp.1108–1116.
- Sigal, I., Yang, H., Roberts, M. D., Downs, J. C., 2010. Morphing methods to parameterize specimen-specific finite element model geometries. *Journal of biomechanics*, 43(2), pp.254–62.
- Sigal, I., Hardisty, M.R. & Whyne, C.M., 2008. Mesh-morphing algorithms for specimen-specific finite element modeling. *Journal of biomechanics*, 41(7), pp.1381–9.
- Silver, F., Christiansen, D. & Buntin, C., 1989. Mechanical properties of the aorta: A review. *Crit Rev Biomed Eng*, 17(4), pp.323–58.
- Sollie, H., 2002. Automatic segmentation and registration of CT and US images of abdominal aortic aneurysm using ITK, Norwegian University of Science and technology.
- Taylor, C. a & Figueroa, C. a, 2009. Patient-specific modeling of cardiovascular mechanics. *Annual review of biomedical engineering*, 11, pp.109–134.

- Thompson, P.M. & Toga, a W., 1997. Detection, visualization and animation of abnormal anatomic structure with a deformable probabilistic brain atlas based on random vector field transformations. *Medical image analysis*, 1(4), pp.271–94.
- Tiilikainen, N.P., 2007. A comparative study of active contour snakes. *Copenhagen University, (DIKU)*.
- Tsamis, A., Krawiec, J.T. & Vorp, D.A., 2013. Elastin and collagen fibre microstructure of the human aorta in ageing and disease: a review. *J R Soc Interface*, 10(83).
- U.S. National Library of Medicine, 2013. The Visible Human Project. Available at: <http://www.nlm.nih.gov/research/visible/>.
- Ukwatta, E., 2013. *Vascular Segmentation Algorithms for Generating 3D Atherosclerotic Measurements*. The University of Western Ontario London.
- Viceconti, M., Bellingeri, L., Cristofolini, L., Toni, A., 1998. A comparative study on different methods of automatic mesh generation of human femurs. *Medical engineering & physics*, 20(1), pp.1–10.
- Vorp, D., 2007. Biomechanics of abdominal aortic aneurysm. *Journal of biomechanics*, 40(9), pp.1887–1902.
- WebMD, 2009. Heart Health Center. Retrieved from: <http://www.webmd.com/heart/picture-of-the-aorta>, [Accessed Decemeber 1, 2013].
- Withey, D.J. & Koles, Z.J., 2007. Medical Image Segmentation: Methods and Software. In *Noninvasive Functional Source Imaging of the Brain and Heart and the International Conference on Functional Biomedical Imaging*.
- Wolf, I., Eid A., Vetter M., Hassenpflug P., Meinzer H., Krebsforschungszentrum D., 2003. Extension of 2D segmentation methods into 3D by means of Coons-patch interpolation. *Medical Imaging*, 5032, pp.1413–1420.

- Wolters, B.J.B.M., Rutten, M.C.M., Schurink, G.W., Kose, U., de Hart, J., van de Vosse, F.N., 2005. A patient-specific computational model of fluid-structure interaction in abdominal aortic aneurysms. *Medical engineering & physics*, 27(10), pp.871–883.
- Wörz, S., Hendrik von T., Verena H, Dittmar B., Hans-Ulrich K, Karl R. ,2010. 3D Quantification of the aortic arch morphology in 3D CTA data for endovascular aortic repair. *IEEE Transactions on Biomedical Engineering*, 57, pp.2359–2368.
- Xu, C. & Prince, J.L., 1998. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing: a publication of the IEEE Signal Processing Society*, 7(3), pp.359–69.
- Zhang, Y. et al., 2012. An atlas-based geometry pipeline for cardiac Hermite model construction and diffusion tensor reorientation. *Medical image analysis*, 16(6), pp.1130–1141.
- Zhang, Y., 2013. Image-based geometric modeling and mesh generation, Springer.XI, X301 p. 133 illus.
- Zhao, F., Zhang, H., Wahle, A., Scholz, T., Sonka, M., 2006. Automated 4D segmentation of Aortic Magnetic Resonance Images. *Proceedings of the British Machine Vision Conference 2006*, 26, pp.26.1–26.10.
- Zitová, B. & Flusser, J., 2003. Image registration methods: a survey. *Image and Vision Computing*, 21(11), pp.977–1000.
- Zohios, C., Kossioris, G. & Papaharilaou, Y., 2012. Geometrical methods for level set based abdominal aortic aneurysm thrombus and outer wall 2D image segmentation. *Computer methods and programs in biomedicine*, 107(2), pp.202–217.