UNIVERSITÀ DEGLI STUDI DI PAVIA

Facoltà di Ingegneria

Dipartimento di Meccanica Strutturale

Transcatheter aortic valve implantation: from clinical data to finite element models

Impianto valvolare aortico transcatetere: dai dati clinici ai modelli di elementi finiti

Supervisor: Professor

Ferdinando Auricchio

Co - supervisor:

Simone Morganti

Author:

Danila Vella

UIN **405345**

Academic year 2012/2013

To my family

Ringraziamenti

Desidero ricordare tutti coloro che mi hanno aiutato nella stesura della tesi con suggerimenti, critiche ed osservazioni: a loro va la mia gratitudine.

Desidero innanzitutto ringraziare l'Ing. Simone Morganti, per la sua disponibilità e pazienza durante tutto il lavoro; il suo supporto e la sua guida sapiente sono stati elementi fondamentali per la realizzazione di questa tesi.

Desidero ringraziare anche il Prof. Ferdinando Auricchio, relatore di questa tesi, per la cortesia dimostratami.

Un ringraziamento va agli amici, in particolare Tamara e Lara, e ai colleghi dell'Università, che mi hanno incoraggiato o che hanno speso parte del proprio tempo per leggere o discutere con me le bozze del lavoro.

Desidero ringraziare con affetto la mia famiglia per avermi trasmesso serenità, per avermi fornito sostegno, per avermi regalato gioia e amore; tutto questo mi ha dato la forza di affrontare al meglio questi anni di studio.

Un ringraziamento particolare va a Salvatore, per essere sempre presente nella mia vita nonostante tanti sono i chilometri che ci separano.

II

Sommario

Il cuore è l'organo chiave del sistema circolatorio. La sua principale funzione è di pompare il sangue in tutto il corpo. Le valvole cardiache e la parete muscolare assicurano il suo funzionamento. In particolare, la valvola aortica controlla il flusso di sangue dal ventricolo sinistro all'aorta, che è l'arteria principale del corpo. Il sangue ossigenato lascia il cuore attraverso l'aorta, da cui è trasportato verso tutte le parti del corpo.

Dato che l'età media della popolazione è aumentata, le malattie degenerative cardiovascolari sono diventate sempre più frequenti. In particolare, la stenosi aortica è il più frequente disturbo alla valvola aortica, ed è causa di maggiore morbilità e mortalità di altre malattie connesse a questa valvola [1]. La stenosi aortica si verifica quando la valvola si restringe. Questo restringimento impedisce la completa apertura, ostacolando il flusso di sangue dal ventricolo verso l'aorta. Quando si ha la comparsa dei sintomi gravi, l'aspettativa di vita media è di soli tre anni [2]. Questa patologia comprende processi simili a quelli dell'aterosclerosi, compreso l'accumulo di lipidi, infiammazione e calcificazione. La sostituzione della valvola aortica (AVR) è un intervento chirurgico a cuore aperto, raccomandato per la maggior parte dei pazienti che presentano i sintomi patologici e evidenza significativa di stenosi aortica dall'esame ecocardiografico [1]. Alcuni pazienti però, non sono operabili a causa dello stato clinico o di comorbidità. L'impianto valvolare aortico transcatetere (TAVI) rappresenta una nuova alternativa alla chirurgia a cuore aperto per il trattamento della patologia [3]. Questo intervento consiste nell'inserimento di una protesi valvolare attraverso un catetere e l'impianto all'interno della valvola patologica nativa. La TAVI è destinata solamente a quei pazienti che sono ad alto rischio per la chirurgia a cuore aperto. Le protesi attualmente più impiantate sono due: l'Edwards Sapien XT di seconda generazione, che consiste in una valvola in pericardio bovino montata su uno stent in lega cromo-cobalto, e il sistema CoreValve fornito dall'azienda Medtronic, che consiste in una valvola in pericardio porcino montato su uno stent in nitinol. Questa procedura mini-invasiva presenta dei

limiti connessi al design del dispositivo: frattura dello stent, disponibilità per un gruppo limitato di pazienti con anatomia e condizioni molto specifiche e problemi relativi al posizionamento e l'ancoraggio.

Per la pianificazione del trattamento in medicina cardiovascolare e chirurgia valvolare, la diagnosi e l'esperienza rappresentano il paradigma comunemente utilizzato. Fino ad ora, poca attenzione è stata invece rivolta alla possibilità di prevedere l'esito di un'operazione. Il metodo più utilizzato per stabilire se un rimedio medico è opportuno o meno è la statistica, questa scienza però, non può essere usata per predire il risultato di un trattamento su uno specifico paziente. Nuovi strumenti predittivi, che possono essere forniti dalla scienza computazionale, dovrebbero essere sviluppati per supportare l'abilità del chirurgo e le sue competenze. Le tecniche computazioanli attualmente presenti forniscono simulazioni specifiche per paziente, che permettono di valutare l'efficacia di diversi possibili trattamenti e di pianificare e progettare la soluzione chirurgica ottimale [4]. Negli ultimi 30 anni, sono stati utilizzati vari metodi computazionali per studiare la valvola aortica, in particolare la tecnica più usata è l'analisi agli elementi finiti. In questo contesto diverse strategie di simulazione chirurgica sono state sviluppate con l'obiettivo finale di fornire uno strumento per supportare il medico durante il processo decisionale [5].

Per quanto riguarda la TAVI, queste tecniche hanno trovato un'ampia utilità, perché risulta ancora una procedura immatura che presenta alcuni limiti. Inoltre, stabilire il posizionamento ottimale della protesi e scegliere il dispositivo migliore in termini di dimensioni e tipo, è di grande importanza per il successo procedurale e il risultato del paziente. Auricchio et al. [6] hanno presentato una strategia di simulazione per l'impianto transcatetere con dispositivo Edwards SAPIEN, che utilizza un modello aortico specifico per paziente per eseguire l'analisi agli elementi finiti. In questa tesi, partendo da questa strategia, una nuova procedura è stata sviluppata per il pre-processing dei dati degli esami clinici, necessario per eseguire la simulazione. Il lavoro svolto ha permesso di ottenere un modello aortico specifico per paziente e un modello della protesi CoreValve.

Il modello del dispositivo si basa sull'elaborazione dei dati ottenuti da una tomografia computerizzata della protesi e rappresenta uno dei quattro formati attualmente disponibili sul mercato. Per la costruzione della protesi due software sono stati utilizzati: Rhinoceros 5.0 per la costruzione del modello geometrico e Abaqus per quello di elementi finiti. Il modello sviluppato consente l'estensione dei casi clinici su cui può essere applicata la strategia di simulazione.

Il modello aortico è diverso per ogni impianto, a seconda della specifica mor-

fologia del paziente. La costruzione di questo modello ha richiesto l'utilizzo di quattro diversi programmi: OsiriX, Matlab, Rhinoceros e Abaqus. In questo lavoro di tesi, alcuni nuovi algoritmi sono stati implementati per i programmi Matlab e Rhinoceros per rendere il processo di costruzione del modello più veloce, più automatico e applicabile ad un maggior numero di pazienti. Il modello sviluppato include anche la presenza del calcio, un fattore rilevante per la stenosi aortica, che interferisce con il posizionamento della protesi e sul risultato finale dell'intervento. I dati clinici, ottenuti tramite tomografia ed ecocardiografia, utilizzati per progettare la procedura sono stati forniti dall'Istituto Clinico Sant'Ambrogio, con sede a Milano. Due modelli aortici sono stati elaborati applicando la nuova procedura su due casi clinici, dai dati degli esami pre-operativi fino ai modelli di elementi finiti. Inoltre, per verificare l'usabilità dei modelli sviluppati, è stata eseguita la simulazione virtuale completa dell'impianto chirurgico di un paziente.

I risultati suggeriscono che i modelli sviluppati possono essere utilizzati in futuro per eseguire simulazioni virtuali su altri pazienti. Un importante lavoro futuro è la validazione della strategia di simulazione, per cui è necessaria la raccolta di un insieme più numeroso di pazienti e di dati clinici post-operativi. Il lavoro svolto rappresenta comunque un contributo significativo alla fase di elaborazione dei dati clinici necessaria per ottenere la simulazione virtuale della procedura di impianto.

Contents

List of Figures **XII**

1 Introduction **1**

2 Transcatheter aortic valve implant **5**
2.1 The aortic valve 5
2.2 Disease of aortic root 7
2.2.1 Aortic insufficiency 7
2.2.2 Aortic stenosis 8
2.3 Transcatheter aortic valve implant 9
2.3.1 Imaging techniques for TAVI 12
2.3.2 FEA analysys of transcatheter aortic valve implant 14
2.4 TAVI procedure with the CoreValve system 15
2.4.1 CoreValve system 15
2.4.2 TAVI procedure 17
2.5 Collaboration with the *Istituto Clinico Sant'Ambrogio* 18

3 Modeling strategy of TAVI **23**
3.1 Aortic root modeling 24
3.1.1 Step 1: Segmentation of CT data 25
3.1.2 Step 2: Computation of new nodes from STL mesh 27
3.1.3 Step 3: Algorithm of model construction 35
3.1.4 Step 4: Meshing and calcium inclusion 41
3.1.5 Required time 47
3.2 Prosthesis CoreValve modeling 48
3.2.1 Creation of the prosthesis model 48
3.3 Simulation of prosthesis implantation in native aortic root 57

4 Conclusions **65**

A MATLAB codes for aortic model **67**
A.1 Code for STL data elaboration 67
A.1.1 Main function 67
A.1.2 Centerline 72
A.1.3 Nodes elaboration 73
A.1.4 Editing RVB file for model construction 81
A.1.5 Editing script for calcium 92

B MATLAB codes for CoreValve model **95**
B.1 Rotation of elementary unit mesh in a polar series 95
B.1.1 Editing INP file for Abaqus 97

Bibliography **99**

List of Figures

2.1 Longitudinal section of the heart. 5

2.2 Aortic valve from aortic view, the black arrows indicate the leaflet free-edge. 6

2.3 The aortic valve opened with a longitudinal incision; the commissures are represented by the two red lines; the aortic annulus is indicated with the red line. The regions a and b are called interleaflet triangles and the numbers 1,2 and 3 indicate the sinuses. . 7

2.4 The image shows the result of the aortic insufficiency 8

2.5 (a) Transesophageal echocardiograms of a normal aortic valve, axial view; (b) Aortic valve (c) Transesophageal echocardiograms of severe aortic stenosis; the axial view shows diffusely thickened leaflets with a restricted opening motion. (d) Calcified aortic valve 9

2.6 Left: Low-Profile Next-Generation SAPIEN XT Valve; right: CoreValve Medtronic device 10

2.7 This image shows the two most commonly used techniques for TAVI. On the left side we can see two types of implants of the balloon expandable Edwards SAPIEN: trans-femoral (above) and trans-apical (below) access. On the right side we can see two types of implants of the self expandable CoreValve: trans-femoral (above) and left trans-subclavian (below). Image from http://www.cardiachealth.org/ 11

2.8 CT image of the heart 13

2.9 Echocardiography of the aortic root: (a) short axis; (b) long axis 13

2.10 Steps of prosthesis release with Delivery Catheter 16

2.11 Bioprosthesis structure 16

2.12 Superior view of CoreValve, the leaflet is clearly visible 17

2.13 Valvuloplasty 18

2.14 Implantation of the self-expanding aortic valve prosthesis. (a) Pre-interventional supraaortic angiogram; (b) Advancement of the prosthesis over the aortic arch using a snare (*) to facilitate the deflection, the TEE probe is visible; (c) device is positioned within the native valve; (d) Pull back of the outer sheath and deployment of the self-expanding prosthesis (**); (e) Fully expanded valve prosthesis; (f) Final angiogram with no evidence of aortic regurgitation; aortic vein graft anastomses distal to stent prosthesis without flow limitations (arrow) [7]. 19
2.15 The graphic shows the aortic and ventricular pressures, the maximum values do not correspond because of pathology. 21
2.16 Electrode with the external balloon 21

3.1 Workflow 25
3.2 Thorax image from Osirix 25
3.3 Stage 1 26
3.4 (a): STL from first patient; (b): STL from second patient. 26
3.5 Stage 2 27
3.6 Nodes on zx plane 29
3.7 Nodes on zy plane 29
3.8 Centerline for zx plane. Red: five points by user, , Blue: five points computed with exponential model, Green: new points set 30
3.9 Centerline with the normal vectors for each point 31
3.10 Cut plane: new nodes(red), STL nodes (blue), centerline point (black asterisk) 31
3.11 The mapping from cylindrical coordinates to Cartesian coordinates. 32
3.12 Cut plane: filling of 4 slices (new nodes red, STL nodes blue). The centerline node is represented with the black asterisk. 33
3.13 Final model obtained from elaboration of nodes (above: first patient, below: second patient) 35
3.14 Rhinoceros user interface 36
3.15 Stage 3 36
3.16 Echo-cardiography of the aortic valve closed in the diastolic phase, the leaflet free margins are highlight. 37
3.17 Section curves 37
3.18 Inner surface 38
3.19 Aortic surface with three cutting plane for the definition of attachment lines of the leaflets. The nine points are visible. 39
3.20 Edge curves of the leaflets: commissures and attachment lines . . 39

3.21 Surfaces of the leaflets 39
3.22 Complete geometrical model 40
3.23 The obtained geometric models: (a) first patient, (b) second patient. 41
3.24 An example of aortic leaflet mesh 42
3.25 Stage 4 42
3.26 The six junction curves between the aortic root and the leaflets . 43
3.27 Seeding of the leaflets: nodes of the junction curves (fuchsia squares) 44
3.28 (a) Mesh of first patient's model; (b) Mesh of second patient's model 45
3.29 (a)First patient's closed leaflets; (b) Second patient's closed leaflets 46
3.30 Grey: calcium, Blue: closed leaflets. The calcium of the root is also visible in the lower 47
3.31 Required time for each phase of the procedure 47
3.32 Left: CoreValve from STL file, right: CoreValve structure 48
3.33 (a): elementary unit; (b): strip, (c): profile curve 49
3.34 Elementary unit model; in the detail a junction region between the two specular strips 50
3.35 Some extracted junction nodes from the STL mesh of the CoreValve 50
3.36 Addition of two circumferences 51
3.37 Surface 51
3.38 Subdivision in fifteen equal parts (some curves are shown) 52
3.39 Selection of the eleven nodes of junction of a strip 52
3.40 Insertion of two points in a node of junction 52
3.41 Profile curve 53
3.42 Rectangular section curves (cross-sections) 53
3.43 Binaries and the cross-sections 54
3.44 Surface with the elementary unit 55
3.45 (a): the final model; (b): the STL model 55
3.46 Left: profile curve of the final model. Right: profile curve of a previous incorrect model 56
3.47 A detail of the elementary unit mesh, where the junction region is visible 57
3.48 (a) CoreValve mesh; (b) detail mesh; (c) superior view 58
3.49 Abaqus interface 59
3.50 Initial configuration of the CoreValve and the cylinder 60
3.51 Crimping: (a) initial phase, (b) final phase 62

3.52 Deployment, second step: (a) the catheter is inserted, (b) the catheter begins to rise, (c) the catheter rises, (d) the CoreValve is released and fits arterial wall 63
3.53 Crystalline arrangement of SMA in different phases 64

Chapter 1

Introduction

The heart is the key organ in the circulatory system. As a muscular pump, its main function is to propel blood throughout the body. Heart valves and a muscular wall ensure its functioning. In particular, the aortic valve controls the blood flow from the left ventricle to the aorta, that is the main artery of the body. Oxygenated blood leaves the heart through the aorta and is carried to all parts of the body.

As the average age of the population increases, degenerative cardiovascular diseases are becoming more frequent; in particular, aortic valve stenosis is the most frequent aortic valve disorder and leads to greater morbidity and mortality than other cardiac valve diseases [1]. Aortic stenosis occurs when the heart's aortic valve narrows. This narrowing prevents the valve from opening fully, obstructing blood flow from the heart into the aorta. With the onset of severe symptoms the average life expectancy is three years [2]. This pathology includes processes similar to those in atherosclerosis, including lipid accumulation, inflammation, and calcification. The aortic valve replacement (AVR), which is an open heart surgery, should be recommended in most patients when the pathologic symptoms appear and there is evidence of significant aortic stenosis on echocardiography [1]. Some patients are inoperable because of clinical status or comorbidities. Transcatheter aortic valve implantation (TAVI), in which a bioprosthetic valve is inserted through a catheter and implanted within the diseased native aortic valve, is an innovative, successful alternative to open-heart surgery for the treatment of aortic dysfunction [3]. TAVI is intended for those patients who are at high risk for open heart surgery because of comorbid conditions. The two most implanted bioprostheses are currently the second generation Edwards Sapien XT, which consists in a a trileaflet bovine pericardial valve mounted on a cobalt-chromium stent frame, and the CoreValve system by Medtronic, which

consists in a trileaflet porcine pericardial valve mounted in a self-expanding nitinol stent. However, this minimally invasive procedure still presents limitations related to device design: stent fracture, availability to a limited group of patients with very specific anatomy and conditions, and positioning and anchoring issues.

For treatment planning in cardiovascular medicine and valvular surgery, diagnosis and experience represent the current paradigm. Up to now, there has been no effort made to predict the outcome of an operation. Statistics is the principal way to establish whether a specific medical remedy is suitable and appropriate or not, but it is not a reliable predictor of success for individual patients. Surgeon skill and expertise should be supported by innovative predictive approaches provided by computational science. These techniques provide patient-specific simulations, which permit evaluation of the efficacy of various possible treatments and the possibility to plan and design the optimal surgical solution [4]. In the last thirty years, many computational studies have been addressed to investigate the aortic valve, in particular the most used technique is finite element analysis. In this context several strategies of surgical simulation have been developed, with the final goal of providing a tool to support the physician decision-making process [5].

Regarding TAVI, these techniques have found a wide utility, because at present it is still an immature procedure which presents some limits. Moreover, achieving optimal positioning of the transcatheter aortic prosthesis as well as choosing the optimal device in terms of type and size, is of great importance to the procedural success and patient outcome. Auricchio et al. [6] have presented a strategy of simulation for transcatheter aortic valve implant performed with the Edwards SAPIEN device, which uses a patient-specific aortic model to perform finite element analysis. In this thesis work, starting from this strategy, a new procedure has been developed for the pre-processing of clinical exam data, which is necessary before performing simulations, in order to obtain both a patient-specific root model and a CoreValve prosthesis model.

The device model is based on the elaboration of data obtained from a micro computed tomography of the prosthesis and represents one of the four sizes currently available on the market. For the construction of the model two softwares were used: Rhinoceros 5.0 for building the geometrical model and Abaqus for the computation of the finite element model, which has been selected for performing finite element analysis. The CoreValve model allows extension of the clinical cases on which the simulation strategy can be applied.

The aortic model is different for each implant, depending on specific patient

morphology. The construction of the patient model requires the utilization of four different programs: Osirix, Matlab, Rhinoceros and Abaqus. Some new algorithms have been developed for Matlab and Rhinoceros to make the process of patient-specific model construction quasi-automatic and faster, as well as applicable to a wider number of real case. Moreover, the developed aortic model includes the presence of calcium, a relevant factor for the aortic stenotic valve, which interferes with the positioning of the prosthesis and on the final results of the surgery. The clinical data, computed tomography and echocardiography, used to design the procedure have been provided by the *Istituto Clinico Sant'Ambrogio* in Milan. Two complete aortic models have been elaborated by applying the new procedure on two clinical cases from pre-operation exam data to finite element models of the aortic root with the inner valve. Moreover, to verify the feasibility of the developed models, a complete virtual simulation of one patient's surgical implant has been performed.

The results suggest that the developed CoreValve model and the construction procedure of the patient-specific aortic model can be used to perform future virtual simulations of other patients. An important future work is the collection of a larger set of patients and post-operative clinical data for the validation of the simulation strategy of the surgical implant. Therefore, the models and procedures developed represent a meaningful contribution to data pre-processing necessary for obtaining virtual simulation of the implant procedure.

Chapter 2

Transcatheter aortic valve implant

Realistic computer-based simulations of transcatheter aortic valve implant (TAVI) require an understanding of the aortic valve physiology and pathology. In this first chapter, three main themes are reviewed: aortic valve function and structure, associated diseases and options for treatment, with a special focus on TAVI.

2.1 The aortic valve

The aortic valve is set in the heart, between the left ventricle and the aorta. The left ventricle is the heart region of the heart with the aim of pumping the oxygenated blood to the entire body through the aorta. The Figure 2.1

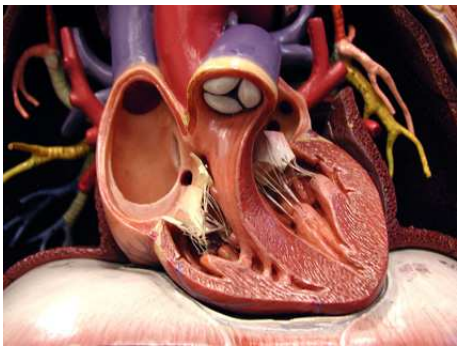


Figure 2.1: Longitudinal section of the heart.

shows the valve position in the heart. The aortic valve opens during the ven-

tricular systole (when the heart contracts for pumping blood), allowing blood flow through the aorta. The valve closes during diastole to prevent blood flow back toward the ventricle, this is the phase of heart relaxing and filling, that comes before the next contraction. The aortic valve consists of three leaflets and sinuses, also called cuspids. The leaflets are the most mobile part of the valve and the sinuses are cavities behind the leaflets. At the lower margin, the sinus become continuous with the left ventricle, and at the upper margin they become part of the ascending aorta. The cuspids represent dilations of the base of the aorta. Apertures of the left and right coronary arteries are present in two of the sinuses, the third is a blind sac. Accordingly, the sinus are named "right coronary" sinus, "left coronary" sinus and "noncoronary" sinus. From the aortic view, the closed leaflets appear to be composed of two parts (see Figure 2.2). One part separates the ventricle from the aorta, bearing the load

Figure 2.2: Aortic valve from aortic view, the black arrows indicate the leaflet free-edge.

of aortic pressure. The second part of each leaflet coapts against the other two leaflets and apparently bears no load, this part is called coaptation surface. The only free boundary of the leaflet, which is visible from the aorta is called the "free edge" of leaflet. The leaflet commissures are formed by the mural regions where two leaflets insert side by side along parallel lines (see Figure 2.3). The sinuses merge with each other at the commissures and continue across the sinus rim distally into the aorta. The inferior part of the leaflet is bounded by the line of attachment, which connects the ventricle with the aorta. It has been reported that the line of attachment in the load-bearing part of the leaflet lies in a plane [8]. The sinotubular junction (STJ) is a circular ring which separates the tubular portion of the ascending aorta from the aortic root. It lies at the level of the commissural apices and provides most of the support for the valve cusps and commissures [8]. The ventriculoaortic junction (AVJ) is the inferior edge of aortic root, this border takes the form of a three-pointed coronet, called the fibrous ring. The planar line, which passes at the sinus bases, represents the ideal ring called *aortic annulus*, routinely measured by imaging technique. The

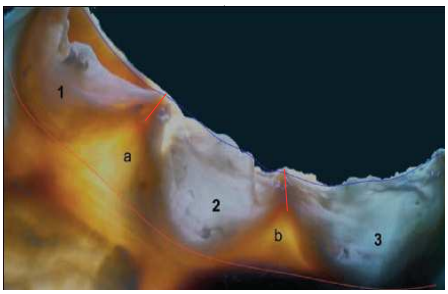


Figure 2.3: The aortic valve opened with a longitudinal incision; the commissures are represented by the two red lines; the aortic annulus is indicated with the red line. The regions *a* and *b* are called inter-leaflet triangles and the numbers 1,2 and 3 indicate the sinuses.

sinuses and leaflets usually differ in size, making the root slightly asymmetrical. Valve anatomy has a direct bearing on valve function. With the growth of surgical procedures and the increasing use of bio-prosthetic valves it becomes important to examine how valve structure relates to valve function. The line of leaflet attachment and the line of leaflet coaptation are the locations where the leaflets experience maximum flexion, that result in highest mechanical stress. Therefore leaflets can be subjected to tissue damage and valvular pathology, which in most cases are characterized by the presence of calcium. Thubrikar et al. studied the sites of calcific deposits and correlated them with the sites of highest mechanical stress in the aortic valve leaflets [8].

2.2 Disease of aortic root

Diseases of the aortic root are intimately related to abnormalities and malformations of the valve. There are several ways in which the aortic valve can become diseased; however, all diseased valves present themselves as being stenotic, incompetent, or both. A stenotic valves offers significant obstruction to the forward blood flow, generally the obstruction results from a narrowed orifice of the valve. An incompetent (regurgitant, insufficient) aortic valve allows blood to flow back into the left ventricle, thereby reducing the net forward flow.

2.2.1 Aortic insufficiency

Aortic insufficiency has been observed as occurring either by itself or in association with aortic stenosis. The causes of this disease are several: postinfilammatory disease of rheumatic origin, aortic root dilatation, incomplete closure of congenital bicuspid valve, infective endocarditis and quadricuspid valve [8]. The

aortic regurgitation may depend either on valve leaflets or on other components of the aortic root, in particular pathologic dilation of the aortic annulus or of the sinuses may cause aortic regurgitation. The consequences of this disease include left ventricle dilation or hypertrophy,including remodeling of the left ventricle [4]. Severe aortic regurgitation generally requires surgical treatments, including substitution of the native valve with biological or mechanical one.

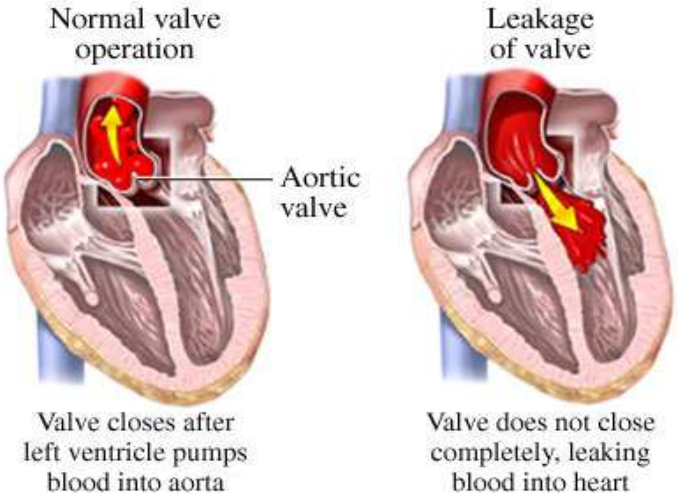
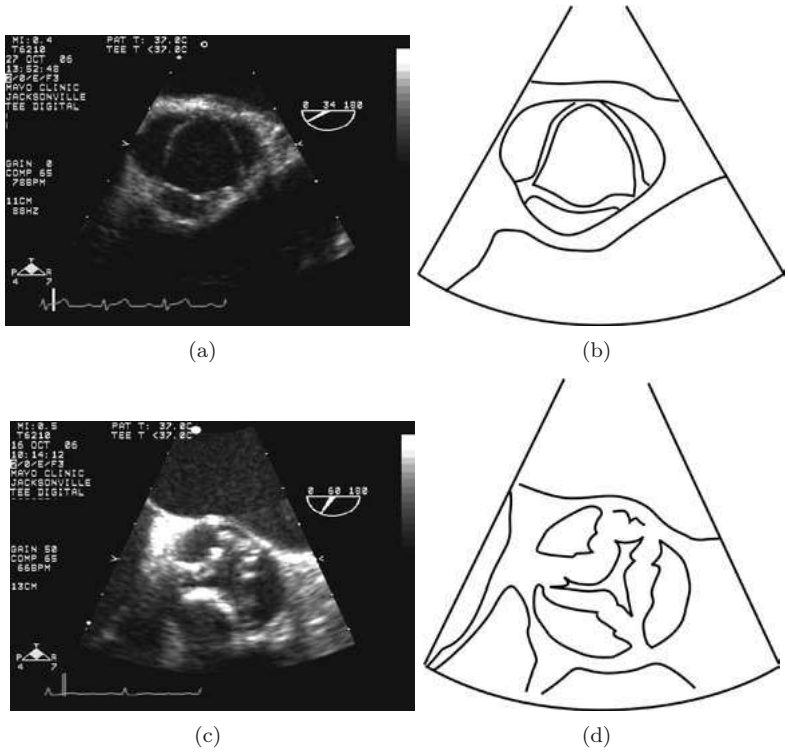


Figure 2.4: The image shows the result of the aortic insufficiency

2.2.2 Aortic stenosis

Aortic valve stenosis affects 3% of persons older than 65 years and leads to greater morbidity and mortality than other cardiac valve diseases [1]. The pathology of aortic stenosis includes processes similar to those in atherosclerosis, including lipid accumulation, inflammation, and calcification. The most common causes of pure aortic stenosis are calcification of bicuspid valve(which consists in the fusion of two leaflets), commissural fusion, degenerative calcification of tricuspid valves, sinuses fibrosis and calcification of rheumatic origin. The development of significant aortic stenosis tends to occur earlier in those with congenital bicuspid aortic valves. Although the survival rate in asymptomatic patients is comparable to that in age- and sex-matched control patients; survival notably worsens after symptoms appear. Aortic valve replacement should be recommended in most patients with any of these symptoms accompanied by evidence of significant aortic stenosis on echocardiography. Aortic valve replacement is the only effective treatment for hemodynamically significant aortic stenosis. The surgery has an average perioperative mortality rate of 4% and a

risk of prosthetic valve failure of approximately 1% per year [1].



(a) (b)

(c) (d)

Figure 2.5: (a) Transesophageal echocardiograms of a normal aortic valve, axial view; (b) Aortic valve (c) Transesophageal echocardiograms of severe aortic stenosis; the axial view shows diffusely thickened leaflets with a restricted opening motion. (d) Calcified aortic valve

2.3 Transcatheter aortic valve implant

Aortic stenosis is an insidious disease with a high rate of death (approximately 50% in the first 2 years after symptoms appear) among untreated patients [2]. Surgical replacement of the aortic valve reduces symptoms and improves survival in patients with aortic stenosis, and in the absence of serious coexisting conditions, the procedure is associated with low operative mortality. However, in clinical practice, at least 30% of patients with severe symptomatic aortic stenosis do not undergo surgery for replacement of the aortic valve, due to advanced age, left ventricular dysfunction, or the presence of multiple coexisting conditions. For these patients, who are at high surgical risk, a less invasive treatment may be a worthwhile alternative [9].

Transcatheter aortic-valve implantation (TAVI) is a new procedure, in which a bioprosthetic valve is inserted through a catheter and implanted within the diseased native aortic valve [10]. A percutaneous alternative was first explored in an animal model by Andersen et al. [11]. Subsequently, a number of groups pursued various approaches to transcatheter aortic valve implantation. In 2002 the feasibility of percutaneous AVR was demonstrated in humans by Cribier et al [12], that performed the first clinical implant of a percutaneous balloon-expandable aortic valve at the level of the native valve. In 2004, Grube implanted for the first time a self-expandable transcatheter aortic valve [4, 13]. Since 2002, when the procedure was first performed, there has been rapid growth in its use throughout the world for the treatment of severe aortic stenosis in patients who are at high surgical risk. The most recent clinical studies show that the rate of death from any cause at 1 year among patients treated with TAVI was approximately 25% [10]. On the other hand the TAVI procedure is still immature due to limited follow-up data and durability evaluation. Conventional open heart surgery remains first-line therapy for symptomatic aortic stenosis. However, percutaneous valve replacement is a viable alternative to conventional open heart surgery in selected high-risk patients with severe symptomatic aortic stenosis [14]. The two transcatheter devices currently most used are Edwards SAPIEN, which consists of a cobalt-chromium alloy balloon expandable stent and CoreValve which consists of a nitinol self expandable stent (see Figure 2.6 and Figure 2.7).

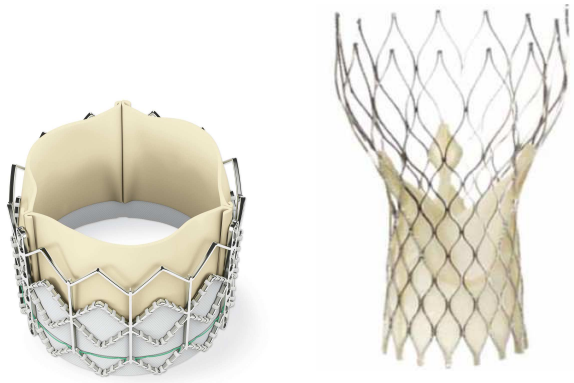


Figure 2.6: Left: Low-Profile Next-Generation SAPIEN XT Valve; right: CoreValve Medtronic device

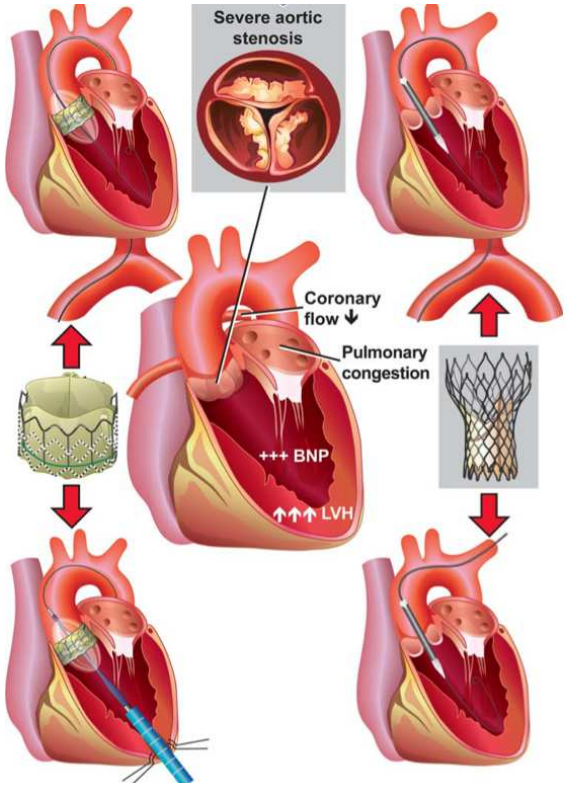


Figure 2.7: This image shows the two most commonly used techniques for TAVI. On the left side we can see two types of implants of the balloon expandable Edwards SAPIEN: trans-femoral (above) and trans-apical (below) access. On the right side we can see two types of implants of the self expandable CoreValve: transfemoral (above) and left trans-subclavian (below). Image from http://www.cardiachealth.org/

2.3.1 Imaging techniques for TAVI

Preprocedural assessments of the patient's iliofemoral system, aorta, and stenotic valve are necessary not only for patient selection (inclusion and exclusion criteria), but also for preprocedural planning of access approach (e.g., transfemoral or transapical), valve sizing, and deployment [15]. The optimal valve function relies on, among other things, accurate sizing that is the selection of prosthesis size to match patient anatomy. Annular sizing of the aortic valve is essential to TAVI success regardless of the access whether femoral, subclavian or transapical is used. Current recommendations and clinical practice require patient eligibility for transcatheter valve therapy and prosthesis sizing to be largely based on the aortic annulus measurements on TTE and TEE, however CT has an established and expanding role in the setting of patient selection and planning of TAVI [16]. Pre-intervention morphological patient screening could also include other techniques: cardiac magnetic resonance imaging, invasive cardiac evaluation with coronary angiogram and left ventriculography [17]. These exams are necessary for identifying the specific location of the area with reduced luminal size which could interfere with the catheter advancement into the vessel. At the time of device implant, a combination of fluoroscopy, aortography, and echocardiography (transthoracic, transesophageal, and/or intracardiac) are employed [15]. In this work the clinical data used has been provided by the *Istituto Clinico Sant'Ambrogio* and consists of echocardiography and computed tomography. The procedure that was developed generates a patient specific model based on data from CT of the patient. Moreover, for creation of the inner valve of the model a measurement from echocardiography is necessary.

Computed tomography

A computed tomography (CT) scanner uses X-rays, a type of ionizing radiation, to make detailed pictures of structures inside of the body. In some cases, a dye called contrast material may be used, to see more clearly the areas of interest. The dye makes structures and organs easier to see on the CT pictures. A ring incorporating one or more X-ray sources and opposing detectors is rotated around the patient, producing and after reconstructing into an image the projections from multiple fan beams. The patient is moved axially trough the donut-shaped scanner; during the movement, equally spaced two-dimensional cross sections (slices) are taken which allow a volumetric tridimensional reconstruction. CT measurements of the thoracic aorta should be performed using an electrodecardiogram to synchronize detection with the heartbeat. In this way precise measurements are possible; in particular, information on leaflets

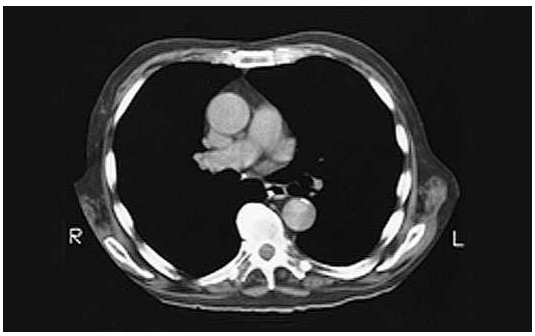
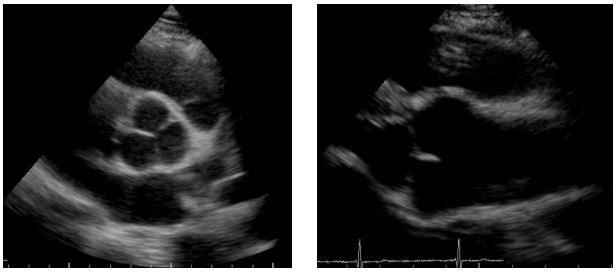


Figure 2.8: CT image of the heart

morphology, symmetry of the sinuses, linearity or tortuousity of the vessel may be obtained [4].

Bidimensional echocardiography

Echocardiography is a methodology which uses standard ultrasound technique to image two dimensional slices of the heart. This method is widely used as diagnostic tool to investigate aortic root pathology. The 2D pictures produced by this exam permit obtaining dimensions and morphology of each component of the aortic root. Echocardiography is the recommended initial test for patients with classic symptoms of aortic stenosis. It is helpful for estimating aortic valve area, peak and mean transvalvular gradients, and maximum aortic velocity. These are the primary measures for assessing disease severity, and they have been well validated compared with measurements obtained with cardiac catheterization [1]. Two main approaches may be adopted to investigate



(a) (b)

Figure 2.9: Echocardiography of the aortic root: (a) short axis; (b) long axis

the aortic root by means of 2D echocardiography: trans-thoracic and trans-esophageal. In the first case the echocardiography transducer (probe) is placed on the thorax of the subject and images are taken through the chest wall. This

is a non invasive technique. In the second case, the probe is inserted through the patient's esophagus [4]. Trans-esophageal echocardiography provides a highly accurate anatomic assessment of all types of aortic regurgitation lesions [18].

2.3.2 FEA analysys of transcatheter aortic valve implant

Percutaneous valve implantation is an innovative, successful alternative to open-heart surgery for the treatment of aortic heart valve dysfunction. However, this minimally invasive procedure still presents limitations related to device design: stent fracture, availability to a limited group of patients with very specific anatomy and conditions, and positioning and anchoring issues. Valve migration and atrio-ventricular block are identified as significant complications of TAVI devices. Incidence of atrio-ventricular block requiring permanent pacemaker implantation after TAVI is generally higher compared to open heart-surgery [19]. Achieving optimal positioning of the transcatheter aortic prosthesis as well as choosing the optimal device in terms of type and size, is of great importance to the procedural success and patient outcome, as the goal is to displace the native valve leaflets and deploy the prosthesis within the native valve annulus. If the prosthetic valve assumes a low position, there is a risk of cardiac arrest or paravalvular regurgitation. Alternatively if the valve is implanted too high there is increased risk of valve embolization and aortic root injury.

Computational simulations together with advanced cardiovascular imaging techniques can be used to help understand these limitations and problems, to guide the optimisation process for new device designs, to improve the success of percutaneous valve implantation and, ultimately, to broaden the range of patients who could benefit from these procedures. In order to explore the feasibility of this intervention in a specific patient, preclinical testing and accurate pre-procedural evaluation are crucial. Engineering approaches may be used to improve procedural safety. By combining high-resolution imaging techniques and Finite Element Analysis (FEA), virtual implantation of such devices is possible in order to understand the interaction of the device with the complex anatomical environment for individual patients.

The FEA represents a numerical technique for finding approximate solutions to partial differential equations, permitting the numerical analysis of complex structures based on their material properties [3]. So-called patient-specific models of medical devices can play an important role in improving cardiovascular interventions. These models are built on the base of patient's CT exam, routinely performed before TAVI. The simulations consider the morphological human variability between subjects, thus they can provide invaluable projections

on the in vivo performance [20]. The simulation of the implant can become a fundamental tool to evaluate, for example, the best positioning before performing TAVI.

2.4 TAVI procedure with the CoreValve system

Replacement of a stenotic aortic valve by transluminal delivery has the purpose of restoring normal function to the patient's deficient aortic valve. One of the two most used prosthesis is CoreValve by Medtronic. The first generation device used for the implant had a sheath size of 24 French. Second and third generations were developed respectively with sheath sizes of 21 and 18 French respectively, increase the number of patients qualified for TAVI. The criteria recommended by Medtronic to perform an implant with the current third generation device are: access vessel capable of accommodating an 18 Fr sheath, aortic valve annular diameter of 20 – 27 mm, as measured by echocardiography and CT angio and ascending aorta diameter of less than or equal to 43 mm. The baseline operative risk of the patients was estimated by the logistic EuroSCORE (European System for Cardiac Operative Risk Evaluation). It works in this way: if a risk factor is present in a patient, a weight or number is assigned, the weights are added to give an approximate percent of predicted mortality. The patient is considered high risk if there is a consensus among an independent cardiologist and cardiac surgeon that conventional surgery would be associated with excessive morbidity and mortality [17].

2.4.1 CoreValve system

The CoreValve system by Medtronic is based on three main instruments:

1. the Delivery Catheter, for introduction and positioning of the bioprosthesis
2. the bioprosthesis CoreValve Medtronic
3. the Disposable Loading System for compressing bioprosthesis into the delivery catheter

The frame is composed of nitinol, an alloy of titanium and nickel; it is made up of a net of rhomboidal cells, obtained with laser cut of a compact nitinol module, from which the useless sections are eliminated. The net presents three regions characterized by different radial force, which perform different functions: inflow region, constrained center and outflow region (see Figure 2.11). The structure is designed to be compatible with the native anatomy of the aortic root and the

Figure 2.10: Steps of prosthesis release with Delivery Catheter

surrounding structures. The internal valve of the prosthesis consists of a single

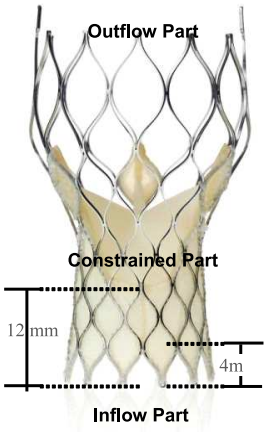


Figure 2.11: Bioprosthesis structure

layer of porcine pericardium sutured to frame with a tri-leaflet configuration. This structure has the function of substituting the native aortic valve. The three leaflets are set in the constrained region, that is positioned above the annulus of the native valve, so the leaflets function is unaffected by annulus shape or dimensions (see Figure 2.16). The inflow region is useful for intra-annulus anchoring, in the inner part a skirt of the same tissue as the leaflets is present, with the function of mitigating paravalvular aortic regurgitation and device migration. Finally, the outflow region provides support to ascending aorta anchoring. The CoreValve prosthesis is available in four different sized models, the measure

Figure 2.12: Superior view of CoreValve, the leaflet is clearly visible

is chosen by physician on the basis of data from the patient's CT, TEE and TTE to better adapt to the specific aortic root. The four available sizes are 23mm, 26mm, 29mm and 31mm, this value indicates the diameter of the inflow tract(see Figure 2.11).

2.4.2 TAVI procedure

The procedure can be performed with different types of access according to the vessel characteristics of the patient: trans-femoral and trans-subclavia (usually left). What follows is a description of the procedure. During the intervention the implant site is visualized by angiography. Usually the procedure is performed with the echo-trans-thoracic (TEE) guidance. Vascular access is obtained either with surgical cutdown of the common iliac artery, the common femoral artery, or the subclavian artery. The procedure is performed with the patient under general anesthesia or with just local anesthesia in combination with a mild systemic sedative/analgesic treatment. The type of hemodynamic support (extracorporeal percutaneous femorofemoral bypass, tandem heart, extracorporeal membrane oxygenation, or none) is left to the discretion of the operator [17]. Balloon valvuloplasty with a balloon under rapid pacing, imposed by a hemodinamic support is performed before device placement. After valvuloplasty, a stiff guidewire placed in the left ventricle is used to pull in the device. After positioning under fluoroscopic guidance the outer sheath is deployed retrogradely and the self-expandable prosthesis is placed. The metallic network of the prosthesis makes possible the visualization through fluoroscopy. If used, extracorporal circulatory support is activated just before device placement across the native valve position and is terminated immediately after withdrawal of the

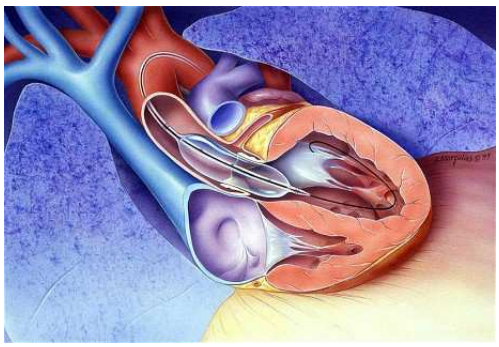


Figure 2.13: Valvuloplasty

delivery catheter and confirmation of adequate valve function. Moreover the use of extracorporal circulation is practiced during the valvuloplasty. Hemodynamic and echocardiographic outcomes are assessed continuously during the procedure [17]. A final supra-aortic angiogram with contrast dye injection is used to reveal if the prosthesis is well-positioned without evidence of aortic regurgitation. Transesophageal echocardiography is used to confirm the device placement and the evidence for significant regurgitation [7].

2.5 Collaboration with *Istituto Clinico Sant'Ambrogio*

The main purpose of this thesis work is to develop a procedure which permits to obtain the complete simulation of a TAVI implant with CoreValve device, using data from the clinical exams performed routinely. Obtaining this simulation before performing the TAVI could be useful for predicting the final result and to provide a support for planning the operation. The design strategy is based on a procedure previously developed by Auricchio et al. [6]. The described procedure consists of modelling the aortic valve and the Edwards SAPIEN prosthesis with the purpose of obtaining final simulation of the implant. In this work this strategy has been re-adapted for the implant with prosthesis CoreValve. The work has been conducted with the main goal of improving the aortic root modelling phase and to automate and accelerate the application of the strategy to specific clinical cases. To achieve this task the use of clinical data was necessary. Therefore a collaboration with the *Istituto Clinico Sant'Ambrogio* was initiated. The Institute is one of the first in Italy for performing transcatheter aortic valve implantations. This collaboration has two main purposes:

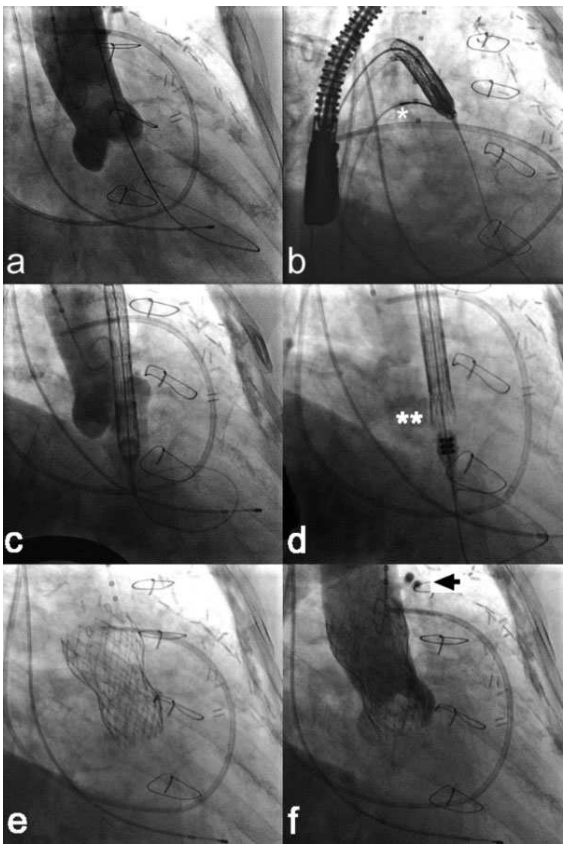


Figure 2.14: Implantation of the self-expanding aortic valve prosthesis. (a) Pre-interventional supraaortic angiogram; (b) Advancement of the prosthesis over the aortic arch using a snare (*) to facilitate the deflection, the TEE probe is visible; (c) device is positioned within the native valve; (d) Pull back of the outer sheath and deployment of the self-expanding prosthesis (**); (e) Fully expanded valve prosthesis; (f) Final angiogram with no evidence of aortic regurgitation; aortic vein graft anastomses distal to stent prosthesis without flow limitations (arrow) [7].

- the observation of the TAVI for better understanding of the real procedure, thus the improvement of the modelling strategy on basis of acquired knowledge
- the collection of data from clinical cases upon which to apply the procedure.

The physician responsible for this collaboration is Dr. Francesco Bedogni, the reference physicians are Dr. Nedy Brambilla for intervention and Dr. Maurizio Tusa for echographic exam. This collaboration made it possible to observe two CoreValve device implants (which for convenience are referred to as first patient

and second patient) and the echocardiography trans-thoracic (ETT) and trans-esophageal (ETE). In particular, the following data were collected:

1. echographic data of the first patient;
2. CT data of the first patient;
3. CT data of the second patient.

During the echocardiography the principle measures taken by the physician are the aortic root diameters corresponding to the regions here listed from bottom to top: the left ventricle outflow tract (LVOT), the annulus, the sino-tubular junction and above the sinotubular junction. In particular the size choice of the size of the CoreValve prosthesis to implant is based on evaluation of these data and on the correspondent measures from the CT exam. The measure of LVOT tract determines for the choice of the device CoreValve or Edwards SAPIEN device for the implant.

For the first patient the access was from the right subclavia. This type of access is used only when the other accesses are impracticable because of the presence of calcifications or of the shrinkage of the diameter of the vessel through which the catheter must pass. The most frequently used accesses are the left subclavia and transfemoral, in these cases the catheter enters the aortic root from a similar angle, this makes the management of prosthesis positioning easier. Conversely, using the right subclavian access the catheter enters from a different angle that makes implantation more difficult. The intervention of the first patient presented complications due to the difficulties linked to access and the critical state of the patient's health. Nevertheless the outcome was positive, though the implant of a pacemaker was necessary after the TAVI. For the second patient the access was of the transfemoral type. The implant was less complicated and the result was positive, however a post-implant valvuloplasty was necessary. The observation of these two implants permitted collection of some information on procedural details in particular pressure monitoring and heartbeat control.

Pressure monitoring

During the intervention the blood pressure in the left ventricle and in the aortic valve is continuously monitored, through a sensor inserted with the catheter. In a healthy patient the maximum value of the aortic and ventricular pressures correspond, while in a patient with a pathological valve these values are different. Because of dysfunction, the maximum aortic pressure is inferior to the maximum

ventricular pressure. The comparison between pressure values before and after intervention is useful in evaluating the result of the prosthesis implant.

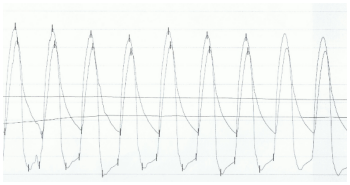


Figure 2.15: The graphic shows the aortic and ventricular pressures, the maximum values do not correspond because of pathology.

Heartbeat control

During the procedure heartbeat is monitored through an electrode with function of pacemaker. The electrode is set at the extremity of the catheter with which it is inserted in the ventricle, the electrode. It is enclosed in a balloon which allows the electrode to bounce into the ventricle. The monitoring of heartbeat

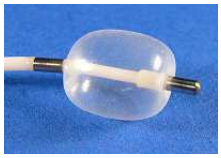


Figure 2.16: Electrode with the external balloon

is very important during the valvuloplasty, in which a balloon is inflated into the aortic root. This technique has the function of tearing the leaflets to make them more weak and flexible for a successful positioning of valvular device. Moreover, it contributes by limiting the negative effect of calcium as regards the adherence between the metallic network of the device and the aortic wall. During the expansion of the balloon a non physiological pressure is exercised in some regions of the heart where the nerves which control the heartbeat are present. This pressure could cause severe injuries if the heart were to maintain a normal heartbeat during valvuloplasty. To avoid this the pacemaker is used to impose a cardiac rhythm of very high frequency during this phase.

Chapter 3

Modeling strategy of TAVI

New developments in cardiothoracic surgery have led to innovative, minimally invasive devices for the treatment of aortic stenosis in patients associated with potential high surgical risk. The two transcatheter devices currently most frequently used are Edward SAPIEN and CoreValve prostheses.

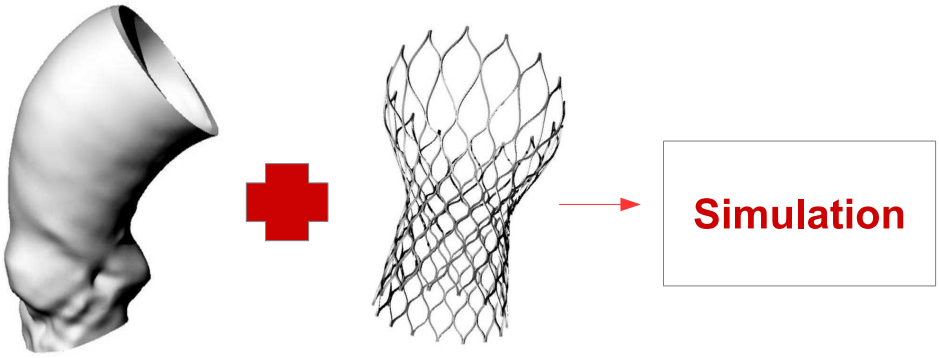
In the last decade, many studies have demonstrated that finite element analysis (FEA) may be successfully used in the field of biomechanics to predict the performance of cardiovascular prosthetic devices implanted in patient-specific geometries [4]. In the article by Auricchio et al. [6] a novel strategy is presented for obtaining the entire Edward SAPIEN prosthetic valve implant in a patient-specific aortic root created by processing medical images. The developed simulation strategy represents a first step towards virtual planning of TAVI procedures aiming at improving the efficacy of the surgical technique and supporting device optimization. To obtain a virtual simulation of the implantation of a specific clinical case two main elements are necessary: the patient-specific aortic root model and the prosthesis model. The goal of this thesis work is to develop a new strategy based on that reported in the article cited above, with two main purposes:

1. to standardize, automate and accelerate the construction strategy of the patient-specific aortic model

2. to increase the clinical cases on which to apply the simulation procedure

To reach these goals two objectives have been pursued:

1. editing of new algorithms for the execution in the Matlab and Rhinoceros programs



2. construction of a prosthesis CoreValve model, from data of the device computed tomography to final mesh

The collected data, thanks to the collaboration with the Istituto Clinico Santo Ambrogio, have been used as reference for building procedure. Moreover, a patient-specific aortic model has been built for each patient using the newly developed procedure, starting from computed tomography data through to the final element model. Finally, to verify the feasibility of the developed models, a complete virtual simulation of one patient's surgical implant has been performed.

3.1 Aortic root modeling

The process of aortic root modeling, which includes the inner aortic valve, is composed of several stages. In each of them a different software is used to elaborate an input document and to produce an output file, that is used for the next step.

- Step 1: segmentation of patient's CT data through the Osirix software to extract aortic root and the calcium in the form of STL file;
- Step 2: elaboration of data with a Matlab code (see A.1): elimination of the inner region of the aortic root, regularization of nodes of STL model and editing a text file with RVB extension, which contains the Rhinoceros 5.0 commands to build the geometric model
- Step 3: running of the RVB file on Rhinoceros 5.0, building of the aortic leaflet through user's interaction and automatic generation of a complete geometric model

- Step 4: definition of the model mesh, simulation of leaflet closure (because the CT data corresponds to a diastolic phase) and inclusion of calcium plaque into the model

Figure 3.1 shows the four stages with corresponding software and input/output files.

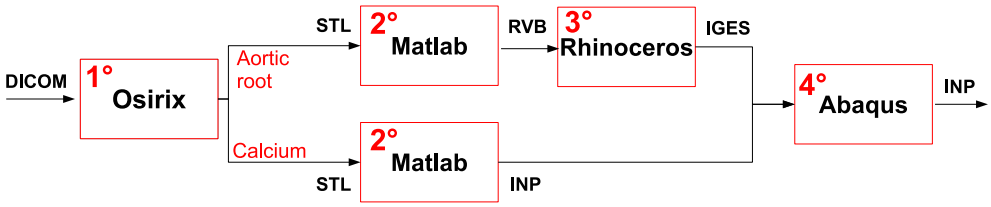


Figure 3.1: Workflow

3.1.1 Step 1: Segmentation of CT data

The CT data appear as files with DICOM extension that corresponds to Digital Imaging and Communications in Medicine. This is a standard for handling, storing, printing, and transmitting information in medical imaging. These files represent the input for Osirix software. A series of slices are recorded from CT, each displays a transversal section of the body part of interest in the form of a greyscale image. Usually these files contain the CT data of the entire thorax with the all inner organs (see Figure 3.2). Through Osirix user interface using *a 3D Surface Rendering* setting it is possible to extract the aortic root and its calcium deposits.

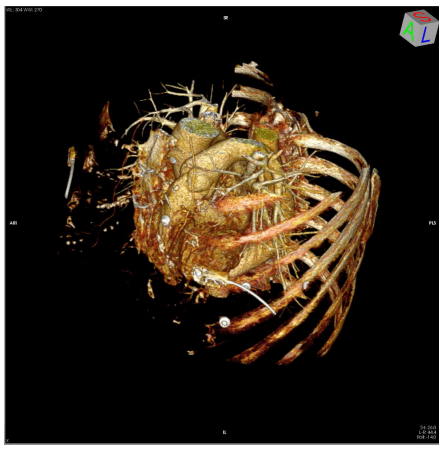
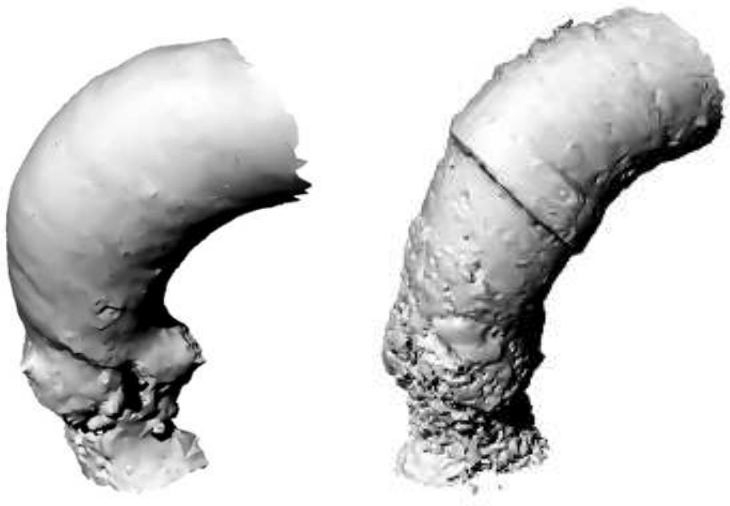


Figure 3.2: Thorax image from Osirix

This operation produces two files with STL extension, one for the aortic root and one for the calcium.

Figure 3.3: Stage 1

The STL format represents a mesh of the object made of triangles, which is not analysis-suitable. The file contains a list of vertex coordinates of triangles. This step requires the user's interaction. The time spent ranges from one to three hours, depending on Dicom data quality. In the analyzed cases the elaboration of the second patient is more difficult than that of the first patient. Patients



(a) (b)

Figure 3.4: (a): STL from first patient; (b): STL from second patient.

undergoing this intervention usually have a calcified valve, due to the advanced age and associated valve dysfunction. The calcium is located generally in the leaflet region. The presence of calcium is relevant for the result of the implant as it interferes with the positioning of the prosthesis. For this reason the calcium is included in the model.

3.1.2 Step 2: Computation of new nodes from STL mesh

The nodes of the STL file of the the aortic root are subjected to several trasformations, which can be summarized in: traslation and rotation, selection and regularization. The selection permits elimination of the inner nodes of the root, their presence is due to non optimal distribution of the contrast medium for the CT exam. The regularization of the nodes is useful for decreasing the noise caused by the low quality of CT data and the variability of the segmentation phase. Before this thesis work, the two elaborations were executed separately, the selection phase through the Abaqus software, requiring the user interaction, and the regularization phase with a Matlab algorithm. To automate the selection phase a new Matlab algorithm has been developed. Currently the two phases are integrated in a single algorithm (see A.1.3). The obtained nodes are used to write a file containing Rhinoceros commands to build a geometric model of the aortic root (see A.1.4). When the Matlab algorithm execution ends this file is saved in a folder as a text file.

The Matlab code accepts as input also the STL file of the calcium, the nodes of the mesh are subjected to traslation and rotation, the same performed for the aortic root. Eventually a text file with triangular-element mesh is written for exporting the calcium geometry in Abaqus (see A.1.5). This file is saved in a folder when the execution ends, before importing into Abaqus.

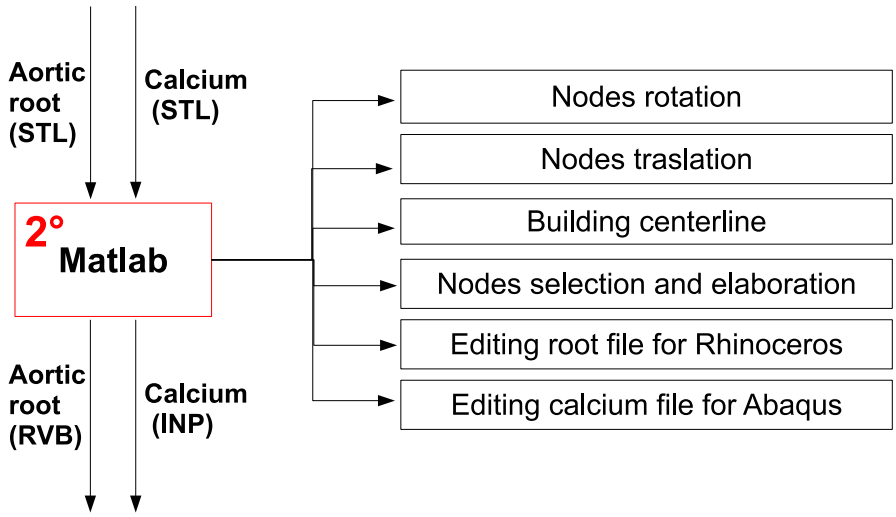


Figure 3.5: Stage 2

The Matlab algorithm carries out these operations:

1. loading nodes from STL file of the aortic root and calcium onto Matlab
2. translation of calcium and root nodes

3. rotation in the zx and zy plane of calcium and root nodes (the two rotation angles are pointed out by the user through mouse clicks on a graphic displaying root nodes in the plane)
4. building a centerline of the root structure
5. selection of the root nodes of interest and computation of new nodes
6. editing a file as input for Rhinoceros 5.0
7. editing a file for the calcium model as input for Abaqus

This algorithm uses two important parameters:

- dl for the computation of transversal section number of the root structure (the higher the dl value, the lower the section number)
- nTh is the number of slices of each transversal cross section

In this work, we mainly focused on the development of more robust features and algorithms of an existing code, especially regarding points 4.,5. and 6 of the supra mentioned list.

Building the centerline (point 4.)

The centerline is built in three steps (see A.1.2):

1. selection of five points which indicate the centerline, by the user on a figure showing nodes in the zx plane (see Figure 3.6); elaboration to obtain a larger points set
2. selection of five points which indicate the centerline, by the user on a figure showing nodes in the zy plane (see Figure 3.7); elaboration to obtain a larger points set
3. computation of (xc,yc,zc) coordinates of a points set which defines the centerline of the aortic root structure

At point 1. two elaborations are performed. The first one consists in fitting the five nodes with an exponential model (X = aebx). This model is used to compute a new set of five points. This step is performed only for the nodes in the zx plane, where the aortic arc is visible. The exponential model allows for a reduction in the tortuosity of the centerline and a better approximation of the real shape of the aortic arc. The second one consists in the interpolation of the new five points with a spline function, that is used to compute a larger points set. This second step is also performed on the five points indicated by the user at point 2.

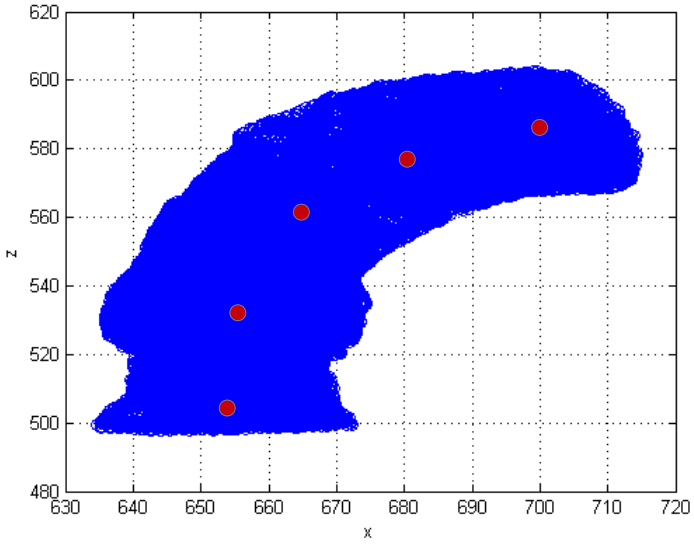


Figure 3.6: Nodes on zx plane

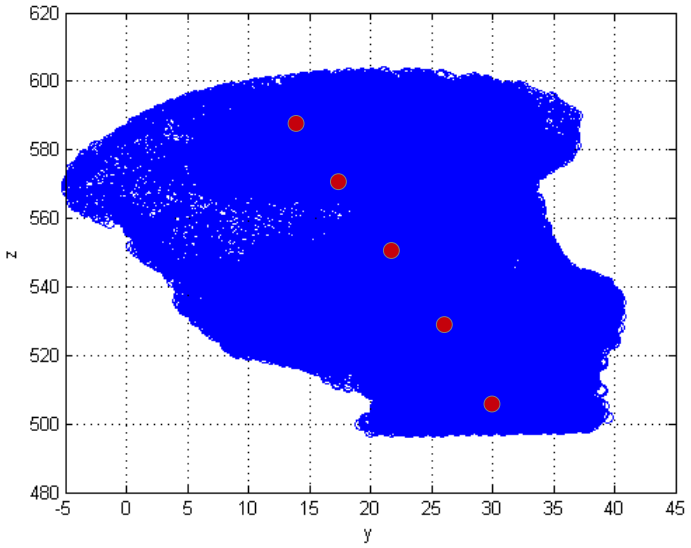


Figure 3.7: Nodes on zy plane

Nodes elaboration (point 5.)

Before this thesis work the procedure for elaboration of nodes was executed in the following steps:

1. insertion by user of the thickness of the arterial wall

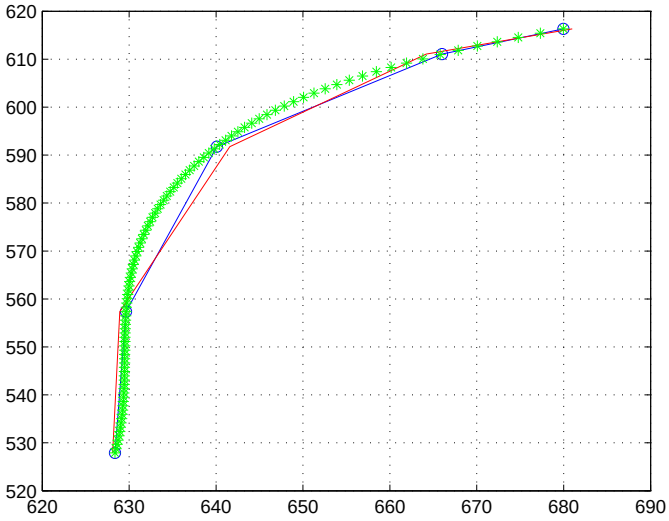


Figure 3.8: Centerline for zx plane. Red: five points by user, , Blue: five points computed with exponential model, Green: new points set

2. definition of a vector for each point of the centerline, which represents the normal vector of the cut plane associated at transversal section
3. selection of a set of nodes associated with the cut plane
4. partition of the set of nodes in nTh slices
5. computation of the mean of nodes into each slice
6. filling the missing slices with new nodes
7. computation of nodes of external arterial wall according to thickness value inserted at step 1.

To improve the procedure, this algorithm was changed at points 2., 4., 6. and 7. The new code is detailed in Appendix A.1.3. Previously the normal vector was provided like the subtraction vector between two adjacent points of the centerline. This method causes the intersection of the transversal section in correspondence of the curve that represents the aortic arc. Because of these intersections the model included only the region of the aortic valve excluding the superior region of the aortic arc. To solve this problem, a new method was introduced to compute the normal vector. Now it is computed as the tangent vector to the centerline curve for each point that defines the curve (see A.1.3). Therefore it is now possible to include the aortic arc in the model.

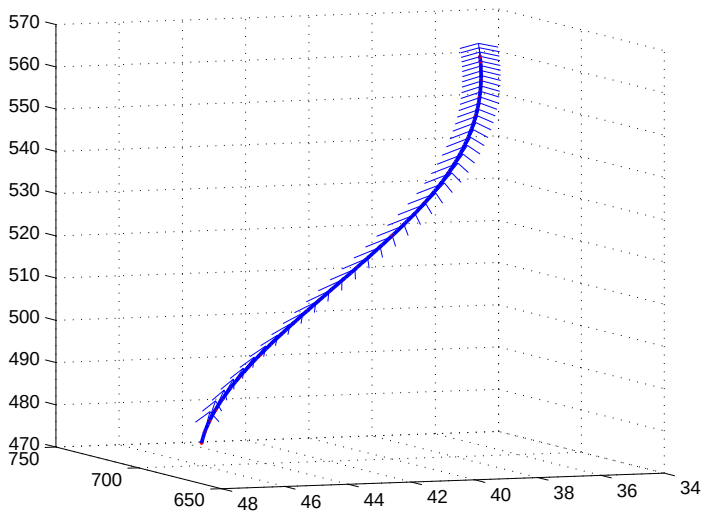


Figure 3.9: Centerline with the normal vectors for each point

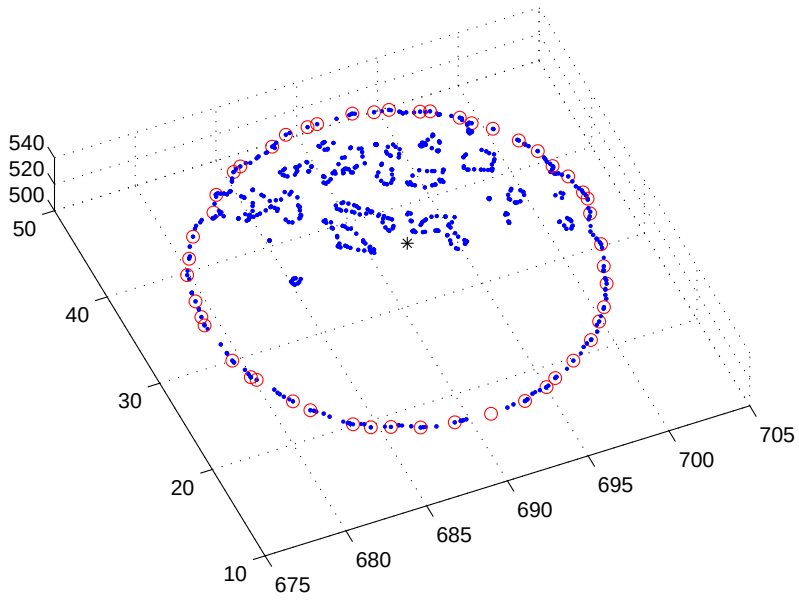


Figure 3.10: Cut plane: new nodes(red), STL nodes (blue), centerline point (black asterisk)

This change required some modifications to the original code, including new algorithms for steps 4., 6. and 7. In points 4. the set of nodes associated to a transversal section is subdivided in nTh slices. In the old code, to perform this operation the system of reference of nodes was transformed from Cartesian to cylindrical. The cylindrical coordinates are *theta, rho* and *z*. *Theta* is a coun-

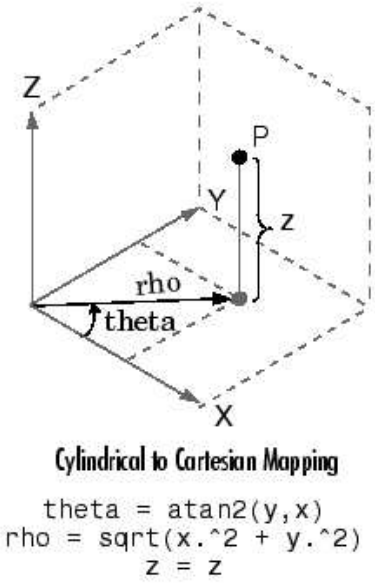


Figure 3.11: The mapping from cylindrical coordinates to Cartesian coordinates.

terclockwise angular displacement in radians from the positive x-axis, *rho* is the distance from the origin to a point in the xy plane, and z is the height above the x-y plane. If the aortic arc is not included in the model, the transversal sections lie in a plane parallel to the xy plane. For this reason the *theta* coordinate of each node is equivalent to the angular position of the nodes in the section. In effect in the old algorithm, the nodes set is subdivided into nTh slices on the basis of the *theta* coordinates of each node. The new algorithm includes the aortic arc, thus the transversal sections do not lie in a plane parallel the xy plane. For this reason the previous method of subdivision does not work in this case. To solve the problem a new function has been introduced (see A.1.3). This function computes the angle between the xy plane and the section plane and uses it to rotate all the section nodes so that they lie in the xy plane. After that the cylindrical coordinates are computed and the nodes set is subdivided into nTh slices.

In point 6. the slices without nodes are filled with new ones, which are computed using the system of cylindrical coordinates (see A.1.3): the z and rho values are computed as the mean between the corresponding values of the nodes in the previous and next slice, and the angle value is that of the corresponding slice into the current section. Regarding point 7., in the existing algorithm the

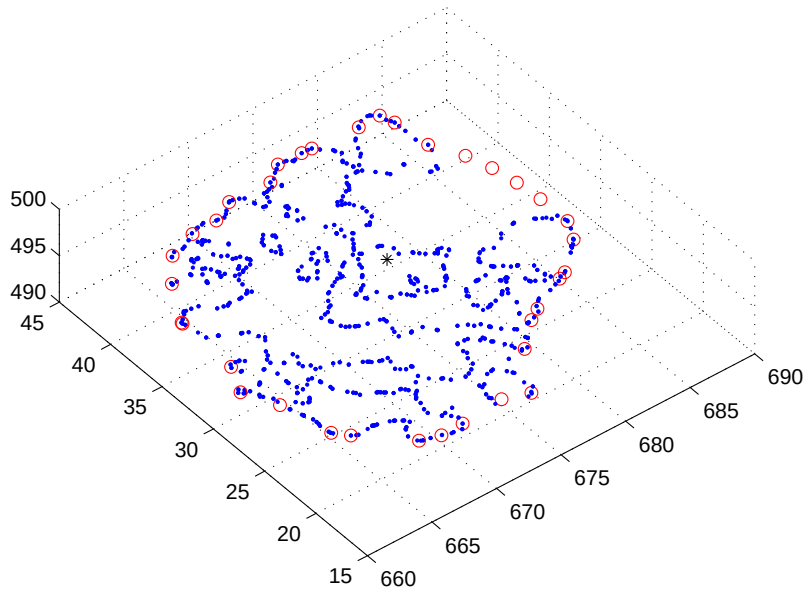


Figure 3.12: Cut plane: filling of 4 slices (new nodes red, STL nodes blue). The centerline node is represented with the black asterisk.

operation was performed using Matlab. However, the inclusion of the aortic arc in the model makes this operation more complex, thus it was decided to implement the whole operation using the Rhinoceros program.

Moreover a new piece of code was introduced to add the operation of removal of the inner root nodes. This algorithm selects a set of nodes in a slice, this is repeated for each slice of each transversal section. The found set is used for computing the mean node according to step 5. of the procedure. The criterion for the selection is based on the distance between each node in the slice and the centerline point associated with the transversal section. For each slice the distance values of the nodes is stored in decreasing order in a vector. The distances are selected with these three steps:

1. computing the differences between adjacent values of the vector; for each computed difference the following control is executed: if the difference

exceeds the assigned value at the parameter *par_ dist*, the two associated distances and the all minor ones will be removed from the vector (see A.1.3)

2. the remaining distances are used to compute the quantile value of *par_ quant* in a previously assigned order, the distances smaller than this value are removed from the vector

3. the remaining distances are used to compute the mean value; on this value the following control is executed: if the difference between the current value and that of the previous slice is greater than the parameter *par_ diff* the current slice is considered empty, otherwise the remaining distances compose the final vector

Finally the nodes selected for the final set are the ones associated at the remaining distances in the already computed vector. In the case of an empty slice the code executes operation 6., described above. The value of the three parameters *par_ diff*, *par_ quant* and *par_ dist* can be changed for adapting the algorithm to the specific aortic root. Recommended values are:

- *par_ diff*: from 3 to 6
- *par_ quant*: from 0.8 to 0.9
- *par_ dist*: 2

Each transversal section with the new computed nodes and the old nodes is displayed during the algorithm execution so the user can evaluate the accuracy of the result and if necessary change the parameter values. Figure 3.13 shows the result obtained applying this procedure to the two studied patients.

Editing file for Rhinoceros (point 6.)

The last step of the Matlab procedure is the creation of a text file for Rhinoceros 5.0 (see A.1.4). This program provides the user with a programming setting called *RhinoScript*. The edited Matlab file is imported in this setting and can be run directly on Rhinoceros. The script is made of several commands that execute geometrical instruction using the nodes produced by the Matlab elaborations. An example of this code is reported below:

```
function write\_script\_rhino3(nz,nTh,Xin,Yin,Zin,name\_file,CL,thick)
fid = fopen(name\_file,'wt');
fprintf(fid,'\%s$\backslash$n', 'Option Explicit');
```

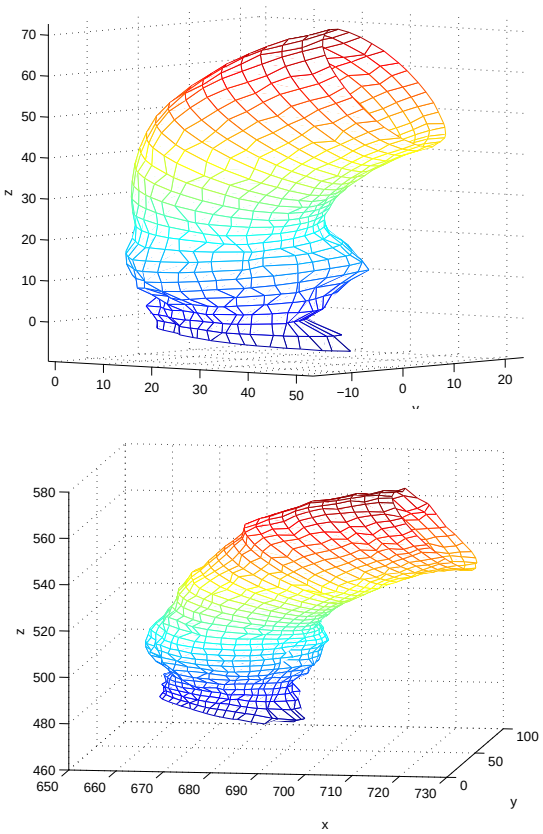


Figure 3.13: Final model obtained from elaboration of nodes (above: first patient, below: second patient)

```
fprintf(fid,'\%s$\backslash$n', 'Call Main()');
fprintf(fid,'\%s$\backslash$n', 'Sub Main()');
fprintf(fid,'\%s$\backslash$n', 'Rhino.RenderSettings 1');
fprintf(fid,'\%s \%d \%$\backslash$n','Dim arrPointCloud(',nTh-1,'),blnCompare');
```

3.1.3 Step 3: Algorithm of model construction

The final geometrical model of the aortic root structure is built in Rhinoceros 5.0. This software is a 3-D free-form modeler, it can be used to create anything type of object. This software makes available an interface with four views (frontal, perspective, left and superior) to display three-dimensional object in a two-dimensional setting and several commands to perform geometric operations. Moreover with the command EditScript can be called a scripting tool based on Microsoft VBScript language. This functionality allows automation of the building of the model. In this case the script is written with a Matlab

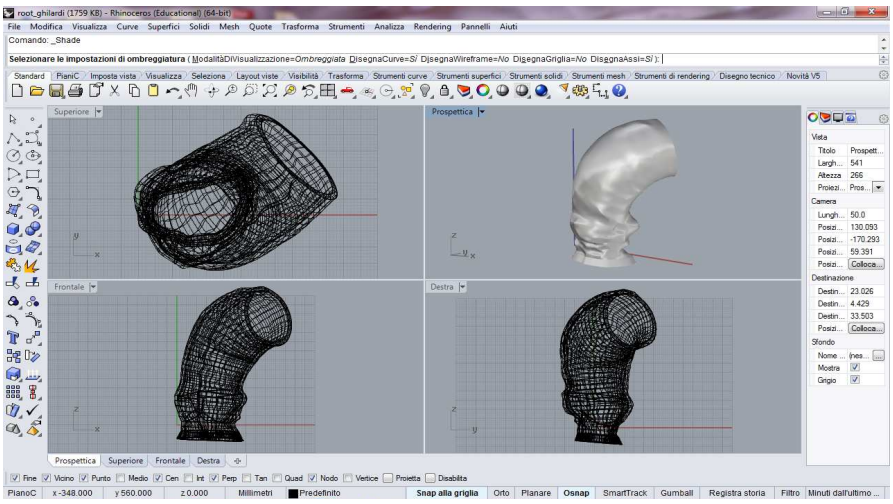


Figure 3.14: Rhinoceros user interface

function and saved with RVB extension. When it is run on Rhinoceros the patient-specific aortic root model is automatically built (see Figure 3.15). The final model is subdivided into two parts: the root and the inner valve, which are saved in two files with IGES extension. In this way the files are ready to be imported into Abaqus for the next process. The script performs two main

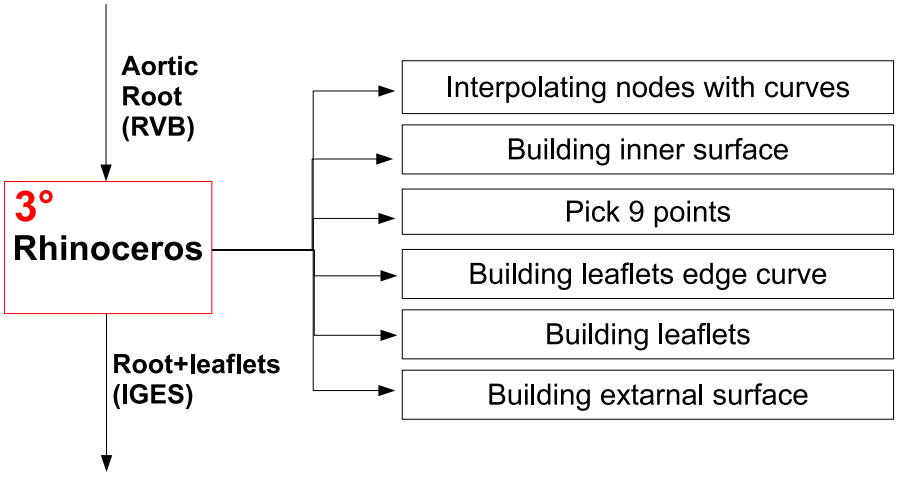


Figure 3.15: Stage 3

operations, the creation of a model of the aortic root and the inner valve. The geometrical instructions for aortic root generation are based on the points processed with Matlab as previously described. These points came from the CT exam of the patient and allow representation of only the geometry of the aortic

root without the inner leaflets. This phase is completely automatic, whereas the participation of the user is necessary for the construction of the inner valve. To generate the inner valve the user defines 9 points, through mouse clicks, on the root surface, these points are used to build the junction curves between the leaflets and the root. Moreover to build the so-called free edge, the user inserts by keyboard the values of the three rays of the arcs which correspond to the superior edge of the leaflets. These rays are computed by a Matlab algorithm that requires as input value the length of the free edge, which can be measured from the patient's echo-transthoracic exam (see Figure 3.16). The main steps

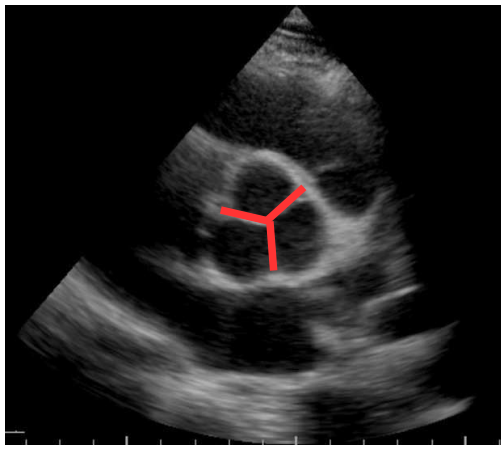


Figure 3.16: Echo-cardiography of the aortic valve closed in the diastolic phase, the leaflet free margins are highlight.

performed by this script are:

1. building of the section curves of the inner surface of the root by interpolation of the nodes (see Figure 3.17)

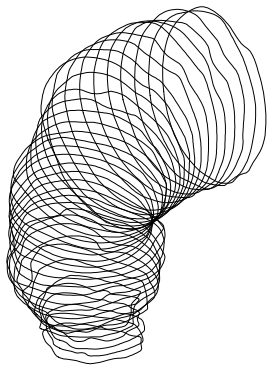


Figure 3.17: Section curves

2. building of the inner surface by loft operation (see Figure 3.40)

Figure 3.18: Inner surface

3. selection of 9 points on surface by mouse click

4. construction of six planes passing through the nine points which cut the root surface (see Figure 3.19), each plane is defined by three points (three planes are used for defining the commissures, and the other three are used for defining the lines of attachment of the leaflets)

5. extraction of the edge curves of the leaflets (see Figure 3.20), as subcurves of the intersections found between root surface and the six computed planes

6. cutting the root surface with the extracted curves

7. building the three free leaflet edges (the user inserts the rays of each corresponding arc)

8. building the leaflet surfaces (see Figure 3.21) from the obtained edge curves

9. computing the nodes of the external surface of the root (the thickness of the aortic wall is defined by user in the Matlab algorithm) and building the section curves by interpolation of these nodes

10. building the external surface of the aortic root by loft operation

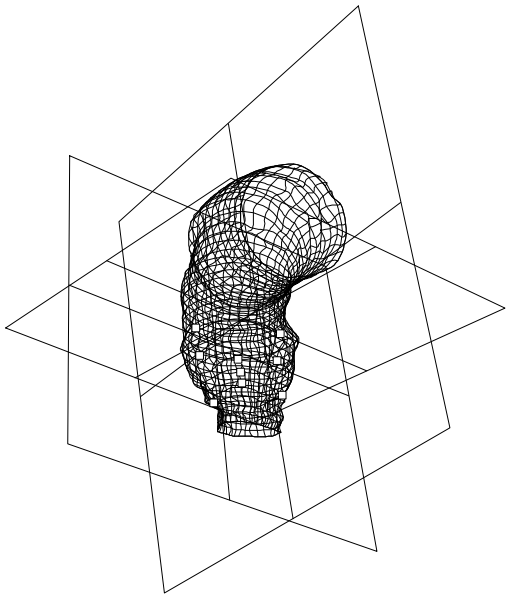


Figure 3.19: Aortic surface with three cutting plane for the definition of attachment lines of the leaflets. The nine points are visible.

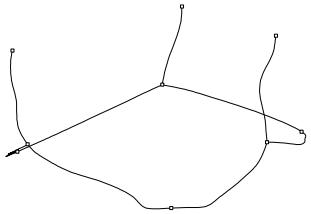


Figure 3.20: Edge curves of the leaflets: commissures and attachment lines

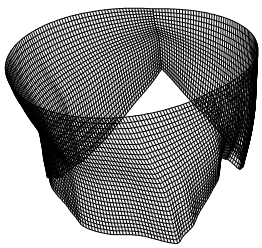


Figure 3.21: Surfaces of the leaflets

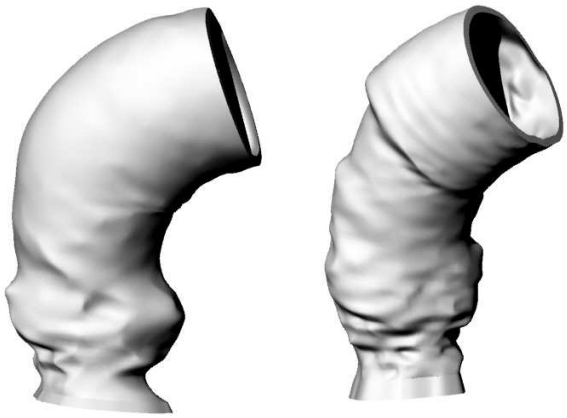
Figure 3.22: Complete geometrical model

As example of the script, the initial part of the function to build the free leaflet edges is reported below:

```
Function build\_arc(arrPt1, arrPt2)
Dim  curve, mid, dist,rag, cat,vect,origin,vectr,ang,plane,arc,p,del
Rhino.Print "Inserire il raggio:"
rag = Rhino.Getreal
curve = rhino.AddCurve(Array(arrPt1, arrPt2))
mid = rhino.CurveMidPoint(curve)
dist = rhino.CurveLength(curve)
If rag > (dist / 2) Then
cat = Sqr((rag \^{} 2) - ((dist / 2) \^{}2))
Else
Rhino.MessageBox "Valore del raggio troppo piccolo!"
Rhino.Print "Inserire il raggio:"
rag = Rhino.Getreal
cat = Sqr((rag \^{} 2) - ((dist / 2) \^{} 2))
End If
```

The Figure 3.23 shows the results of the application of this procedure on the two study cases. In both cases a further work is run: the building of the inferior part of the aortic root. In effect, this region is often irregular because of bad definition of contours in DICOM files. In these cases this part is excluded

from Matlab elaboration. This omission could create some problems for model geometry, therefore in these cases this part is built using Rhinoceros commands.



(a) (b)

Figure 3.23: The obtained geometric models: (a) first patient, (b) second patient.

3.1.4 Step 4: Meshing and calcium inclusion

To study and reproduce the aortic valve behavior a commonly used technique is the finite element analyses (FEA). This approach is a numerical technique to provide approximate solutions of partial differential equations (PDE) used to describe the physical phenomena of the object of study. The basic idea of FEM is to divide a continuum body into discrete finite elements connected by nodes [4]. The set of the discrete finite elements and nodes constitutes the so called mesh of the body (see Figure 3.24). The approximate solution of the entire continuum is then obtained from the assembly of all the individual elements and computed by a computer program. In particular, in this work the one used is Abaqus. This software provides to user an interface and several commands to guide him in performing FEA analysis from meshing to simulation. The two IGES files with the geometric model of aortic root and leaflet are imported in Abaqus and integrated with the geometric model of the calcium. All the information necessary for the definition of this finite element model are saved in a final file (with INP extension) (see Figure 3.25). This file can be used as input for future simulation of the transcatheter aortic valve implant. The main operations performed are:

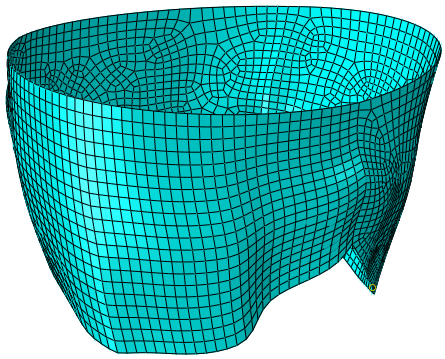


Figure 3.24: An example of aortic leaflet mesh

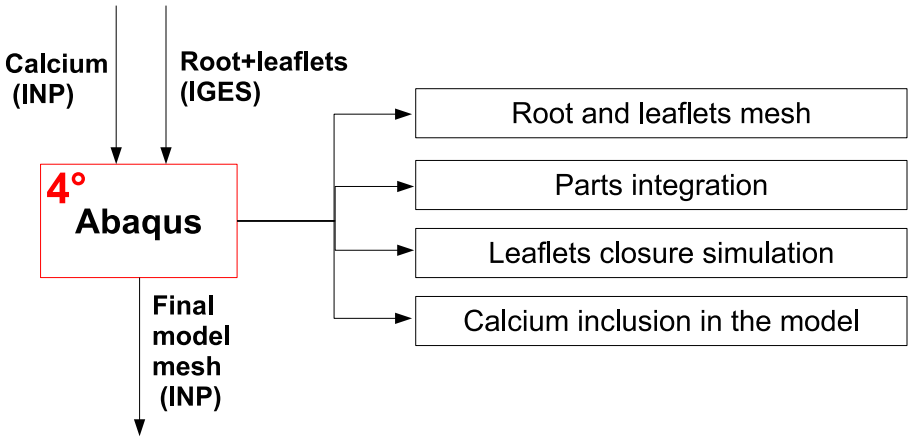


Figure 3.25: Stage 4

1. generation of the mesh of the root and the leaflets;
2. integration of the two parts through a Matlab algorithm;
3. simulation of the leaflets closure;
4. mapping of calcium on the closed leaflets and the root.

In the following, each step is detailed.

Step 1: generation of aortic root and leaflets mesh

In this phase, this operation is performed on both models of aortic root and leaflets. Since the final goal is to obtain the mesh of a single part which include both the root and the leaflet, it is necessary that the two meshes are defined with the same number of elements on the junction curves they have in common. The junction curves are six: three are the basal attachment lines of the leaflets

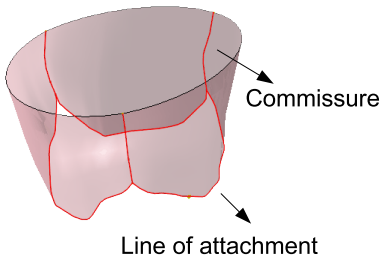


Figure 3.26: The six junction curves between the aortic root and the leaflets

and three are the commissures between the leaflets (see Figure 3.26). The six curves on the root are coincident with the six curves on the leaflets, according to the building criterion. To generate a correct mesh, it is necessary that the mesh nodes of the root curves and the leaflets curves are set in the same points. Before applying a mesh to the geometry two operations should be performed: seeding of the edges and assigning mesh controls. The first operation consists in setting the position of the element nodes on the structure edges, the second consists in the choice of the element shape. In our case the nodes must have the same positions on the junction edges between root and leaflets. The element shape used to mesh the root is hexahedral, because the part is solid, while for leaflets a quadratic shape that generates two-dimensional elements is used.

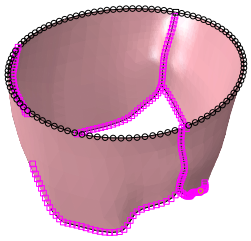


Figure 3.27: Seeding of the leaflets: nodes of the junction curves (fuchsia squares)

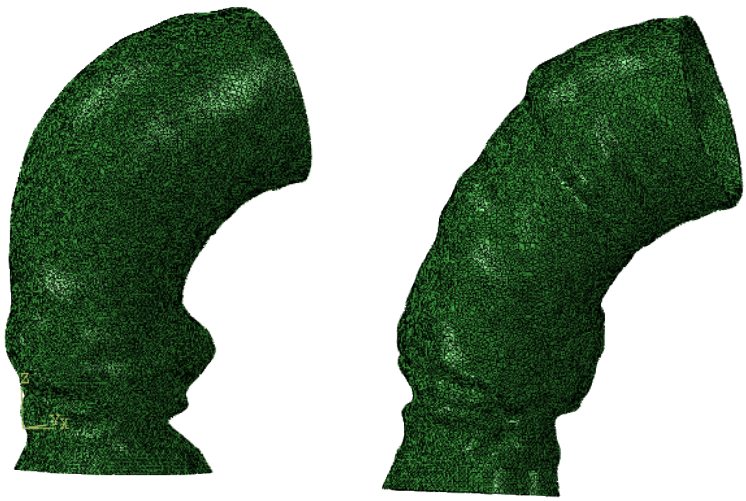
Step 2: integration of the two parts

The root and leaflets meshes are saved in two files, with INP extension, each contains the information about index and coordinates of each node, and index and nodes of each element. As an example, the initial part of the root INP file is reported below, as we can see each geometrical entity is called *part* and is defined by a set of nodes and elements listed in the file.

```
*Heading
** Job name: Job\_ghilend Model name: Job\_root\_ghil
** Generated by: Abaqus/CAE 6.10-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=ROOT
*Node
      1,   11.5236006,  -13.4689646,   17.8124638
      2,    5.8658123,  -7.83258438,   16.5471115
      3,   13.4011707,   14.1615705,   14.7067537
      4,   30.5050545,  -4.35180283,   18.7081947
      5,    46.545681,   11.9986258,   71.6539307
      6,   46.5283661,   11.0768757,    73.408905
      7,   28.2989197,  -2.92127275,   6.25700092
      8,   29.4636631,  -3.49305487,   9.11233997
```

The two files are modified by a simple Matlab algorithm that integrates them in a single file, which contains a single part formed by the union of the two entities.

The result is a single INP file containing the final mesh associated with a single geometrical part (see Figure 3.28).



(a) (b)

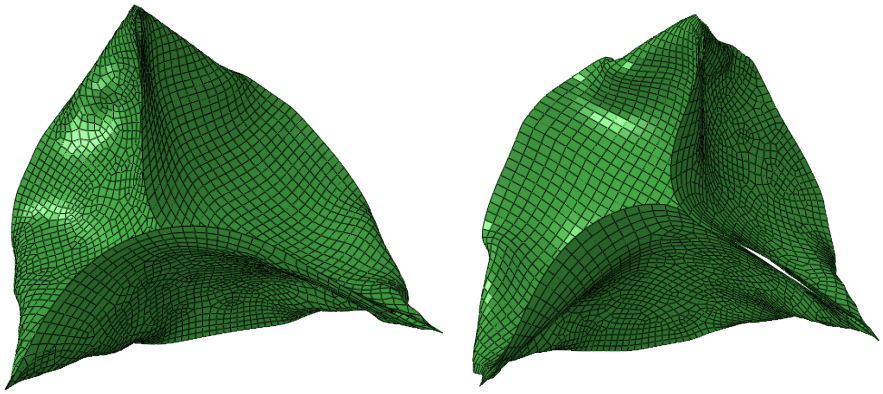
Figure 3.28: (a) Mesh of first patient's model; (b) Mesh of second patient's model

Step 3: simulation of leaflets closure

The obtained model includes the root with the open leaflets. This configuration of the model is partially incorrect because the root model is extracted from the CT data which are recorded when the heart is in phase of diastole, in which the inner valve is closed. For the inclusion of the pathological calcium in the model it is necessary to simulate closure of the leaflets, the calcium indeed is extracted by the same CT exam data used for the root. After simulation, the closed configuration of the leaflets is used to create a map of the calcium regions on three cusps. In the following, the procedure adopted to perform this simulation is described. The leaflet assigned material is elastic isotropic with these characteristics [4]:

1. Density: $1.1 * 10^{-9} Kgl^{-3}$
2. Young's Modulus: 1 Pa
3. Poisson's Ratio: 0.45

The thickness of the leaflets is set of 0.4 mm. A quasi-static analysis is performed using an explicit method in solving partial differential equations. The explicit dynamics procedure is typically used to solve quasi-static simulations involving complex nonlinear effects. The technique of Mass Scaling is used to accelerate the simulation. The principle of this method consists in scaling the mass of the model in order to achieve a larger explicit time-step, thus to accelerate the analysis. The value of the time-step imposed is $1 * 10^{-5}$ sec. This choice is justified by the use of a quasi-static analysis. The contact between leaflets surface during closure is set of tangential and normal type. To determine the closure a pressure load of 0.01 MPa is imposed on external surface of the leaflets. The results are shown in Figure 3.29.



(a) (b)

Figure 3.29: (a)First patient's closed leaflets; (b) Second patient's closed leaflets

Step 4: mapping of calcium on the closed leaflets and on the root

In this stage the calcium is included in the model. This operation is performed separately for root and leaflets. The INP file containing the calcium model, previously elaborated with the Matlab code described in the section 3.1.2, is imported into Abaqus and superimposed on the closed leaflets and on the root. The user selects the leaflets mesh elements that are touched by the calcium, the same action is performed for the root (see Figure 3.30). These elements are saved in two different sets. The two sets are included in the INP file created at step 2, which contains the complete model as a single part. As a result, this final file contains the complete mesh information of the aortic root and the open leaflets with a map of present calcium.

Figure 3.30: Grey: calcium, Blue: closed leaflets. The calcium of the root is also visible in the lower

3.1.5 Required time

Currently, the application of this procedure to build a patient specific model, from clinical data to finite elements model, takes about 4 hours. Most of the time is utilized for the extraction of the root from the CT exam data with the Osirix program and for the Abaqus elaboration. Figure 3.31 shows the required time for each phase of the procedure. The Osirix phase requires the user's interaction.

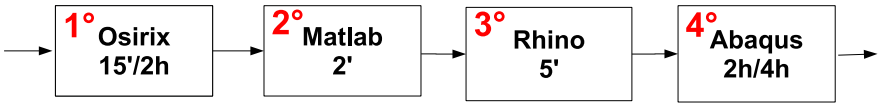


Figure 3.31: Required time for each phase of the procedure

The time spent ranges from 15 minutes to two hours, depending on DICOM data quality and on user's experience with the program use. The production of the geometrical model with Matlab and Rhinoceros algorithms takes about seven minutes in the better case, the most time is for the definition of the 9 points for the building of the leaflets surface. The time necessary depends on the quality of CT exam data. The quality depends on the definition of the edges of the root structure in each trasversal slice of the DICOM file, if they are poorly defined it could be necessary to repeat the procedure. Therefore by changing the parameters it is possible to better match the algorithm to each specific case to obtain an improved geometrical model. Finally, the Abaqus phase can require until 4 hours, the most slow step is the inclusion of the calcium deposits in the model. The time spent depends on user's experience with the program use.

3.2 Prosthesis CoreValve modeling

In the second part of the thesis, a model of the prosthesis CoreValve has been developed. The Edward SAPIEN and CoreValve devices are currently the prosthesis most frequently implanted for the TAVI. Thanks to this new model the strategy of virtual simulation of the TAVI procedure can also be applied to patients which are operated with the CoreValve device. The final aim for this model is that it be utilized as input for the finite element analysis, thus the modeling procedure includes not only the construction of the geometrical model but also the mesh of the model. The geometrical model of the elementary unit, on which the CoreValve structure is based, was built with Rhinoceros 5.0, through the user interface. The operation of meshing of the structure was performed by the Abaqus software. Moreover it was necessary to use a Matlab algorithm to complete the mesh of the model.

3.2.1 Creation of the prosthesis model

The built model represents a CoreValve of size 26 mm, because the two studied surgeries required this measure. The model is based on elaboration of an STL file, generated from a MicroCT of the prosthesis. [1]

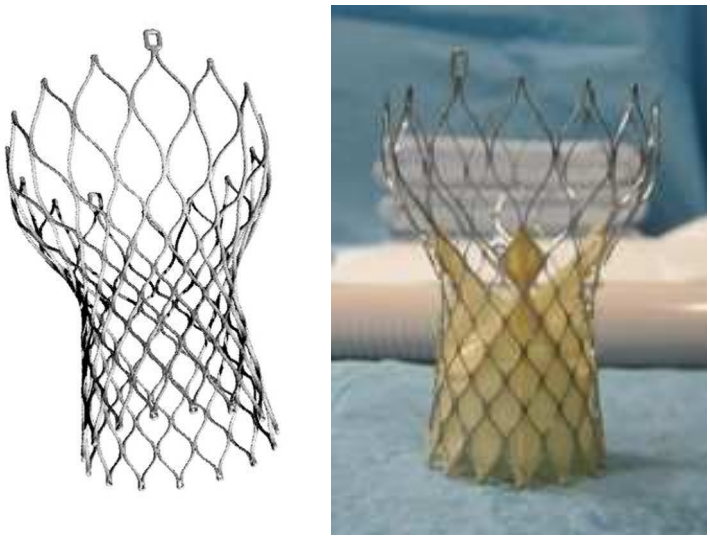


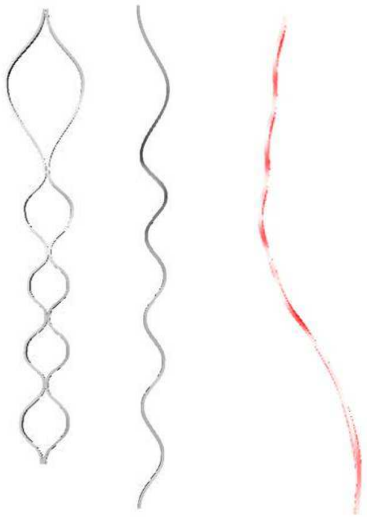
Figure 3.32: Left: CoreValve from STL file, right: CoreValve structure

[1]The aortic prosthesis was provided by the Hospital Cinisello in Pisa by Professor Anna Sonia Petronio, later it was sent to Orthopedic Institute Rizzoli in Bologna to undergo MicroCT to obtain the DICOM data. These data contain only the stent structure, not the soft tissue of the prosthesis, which are not included in the model.

The model was built complying with the following important features of the structure:

1. the prosthesis is made of elementary units repeated in a polar series fifteen times
2. one simple unit consists of two strips with a rectangular cross-section
3. one strip is the mirror copy of the other
4. one strip touches the two near strips in eleven regions, usually called nodes
5. the strip structure not lie in a plane
6. the profile curve of the solid structure of the strip must be smooth, not tortuous

The Figure 3.33 shows some of these features through some images taken from final model. According to these features, these are the steps for building the



(a) (b) (c)

Figure 3.33: (a): elementary unit; (b): strip, (c): profile curve

mesh:

1. construction of the elementary unit geometrical model
2. meshing of this unit and merging of the nodes associated to six junction regions between the two strips (see Figure 3.34)
3. reproduction of this finite element model fifteen times in a polar series

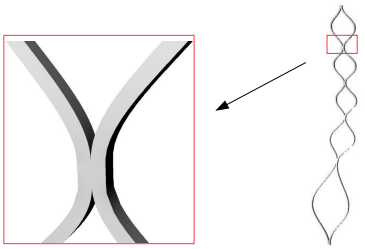


Figure 3.34: Elementary unit model; in the detail a junction region between the two specular strips

4. merging of the nodes associated to all junction regions between the fifteen elementary units

In the following, each step is detailed.

Step1: Construction of the elementary unit geometrical model

The software used to build the geometrical model of the CoreValve elementary unit is Rhinoceros 5.0. The STL file, which contains a triangular-element mesh defining the CoreValve structure, was imported into Rhinoceros. The modeling procedure is totally based on the information provided by this file. The steps performed to build the model are:

1. extraction from the STL file of the points which represent the junctions between all strips of the structure (see Figure 3.35)

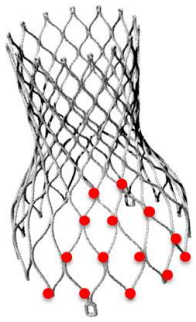


Figure 3.35: Some extracted junction nodes from the STL mesh of the CoreValve

2. generation of eleven circular sections through interpolation of the extracted points that lie on a same plane, translation of each section so that each is aligned, i.e. the centers of each circumference lie on same vertical line

3. addition of two circumferences (see Figure 3.36) in the top and bottom regions at the eleven sections, that are then used to compute the profile curve of the strip

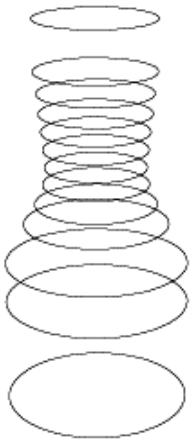


Figure 3.36: Addition of two circumferences

4. construction of a surface (see Figure 3.37) passing through all curves by lofting operation

Figure 3.37: Surface

5. partition of the eleven section curves into fifteen equal parts, in each section the extremes of the obtained sub-curves are aligned with the middle points of the sub-curves on the two adjacent sections (see Figure 3.38)

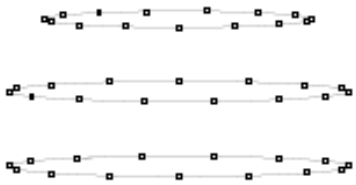


Figure 3.38: Subdivision in fifteen equal parts (some curves are shown)

6. selection of eleven points corresponding to the eleven junction nodes of a single strip (see Figure 3.39)

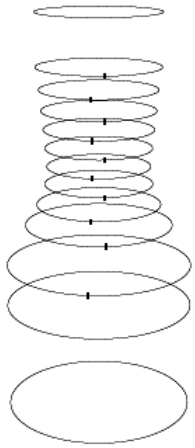


Figure 3.39: Selection of the eleven nodes of junction of a strip

7. insertion of two points for each selected node, which are equidistant from the node and lie perpendicularly at the plane on which lies the section circumference (see Figure 3.40); these points have the function of modeling the regions of the final solid structure corresponding to the junctions with the other strips

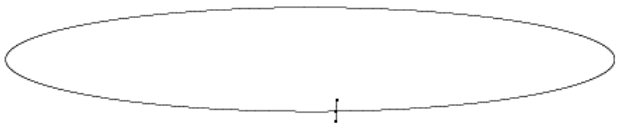


Figure 3.40: Insertion of two points in a node of junction

8. construction of the profile curve (see Figure 3.41) of the strip with the

imposition of two constraints: the curve must lie on the surface and must pass through the all selected points, i.e. the eleven junction nodes and the corresponding two points for each of them

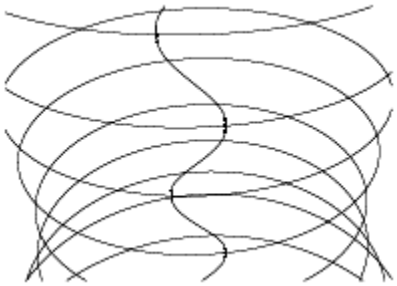


Figure 3.41: Profile curve

9. construction of rectangular section curves (see Figure 3.42) of the strip for each junction node, the segment which merges the two middle points of longer sides of the rectangle lies on the tangential direction to the circumference; the value of the two sides of the rectangle is measured by DICOM data through the Osirix program

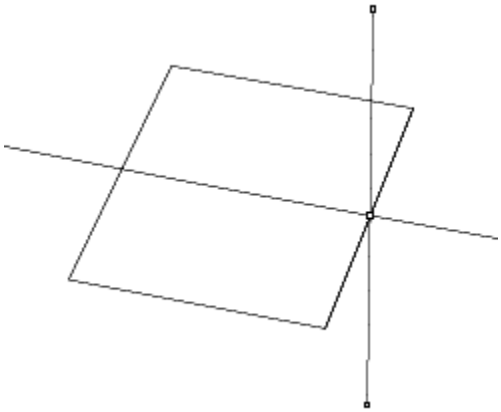


Figure 3.42: Rectangular section curves (cross-sections)

10. construction of two binaries to sweep the cross-section to obtain the solid structure of the strip, obtained from translation of the build profile curve from the junction nodes to two vertexes of each rectangular section (see Figure 3.43)

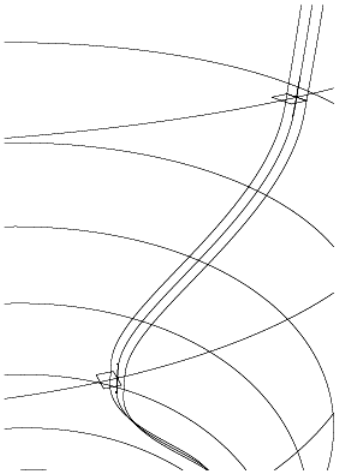


Figure 3.43: Binaries and the cross-sections

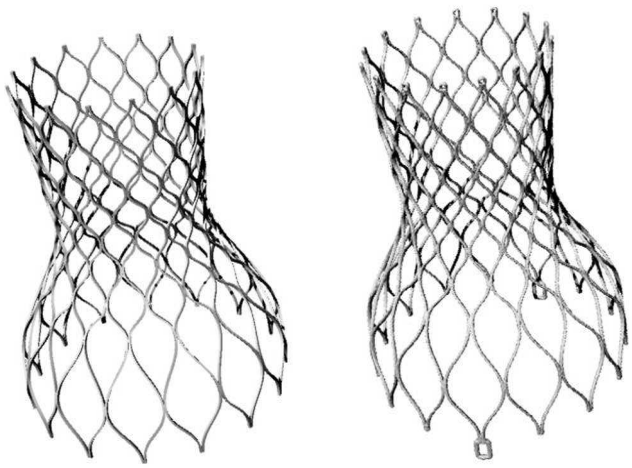
11. building of the solid structure of the strip

12. construction of the specular copy of the strip (see Figure 3.44)

In the following image the geometrical model obtained from development of the elementary unit is presented next to the STL model of the CoreValve. A difficult step of the construction procedure was the computation of the profile curve of the strip. At the beginning, to solve this problem the approach of interpolating the junction nodes of the strip with a curve was used imposing a tangential behavior near the nodes. Rhinoceros software makes available several commands for this operation. Unfortunately the profile curve thus produced does not comply with the feature 6. described above. The resulting strip is shown in Figure 3.46. After several attempts the problem was solved with the introduction of a surface modeled on the transversal circular sections of the CoreValve structure, from which the profile curve is extracted. The idea to use a surface for the curve generation originates from the knowledge of the construction strategy of the real structure. In effect the prosthetic frame (stent) is manufactured by laser cutting of a nitinol metal tube [21].

Another important step of this elaboration was the modeling of the junction regions. The first approach was to represent each junction between the strips with a single point, i.e. the two strips touch themselves in a single point. This solution was abandoned because very high stress is associated with a single point, this could create problems with future simulations. In continuum mechanics, stress is a physical quantity that expresses the internal forces that neighboring

Figure 3.44: Surface with the elementary unit



(a) (b)

Figure 3.45: (a): the final model; (b): the STL model

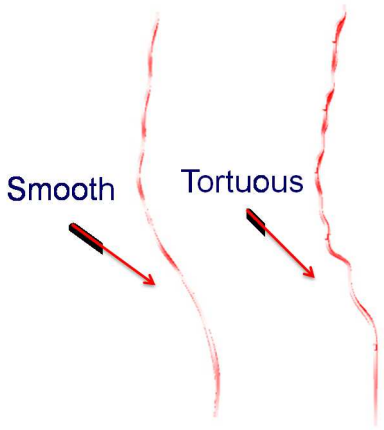


Figure 3.46: Left: profile curve of the final model. Right: profile curve of a previous incorrect model

particles of a continuous material exert on each other. Moreover, the DICOM images show that this model is very different from real structure in which the contact region is more extended than a point.

In a second approach these regions were represented with some parallelepipeds inserted in the strip. In this way the profile curve of the solid structure is made up of different parts: the sides of the parallelepipeds alternate with lines connecting the parallelepiped vertexes. This solution was abandoned because the resulting profile curve was very tortuous and thus does not comply with feature 6, described previously.

In the end, the following choice solved the problem: the profile curve was constrained to intersect three points near each junction region which define the shape of the junction. This operation is detailed at points 7 and 8 of the procedure previously described. The shape of the junctions was completed with a further elaboration performed with the Abaqus software, the merging of the mesh nodes of the two strips in contact associated to these regions.

Step 2: Meshing of the unit model and merging of nodes

The elementary unit is imported into Abaqus as a file with IGES extension and has been discretized using solid tetrahedral elements. The overlapping nodes of the six junction regions, belonging to two specular strips, are joined, this operation is called merging nodes. Finally the mesh information about nodes and elements are saved in a INP file.

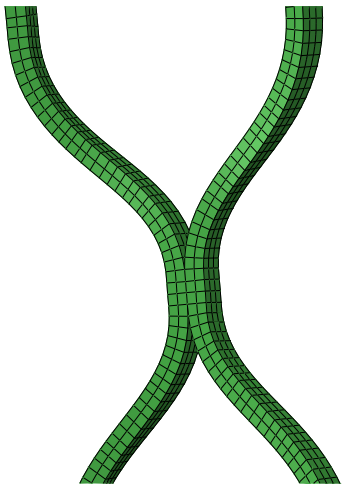


Figure 3.47: A detail of the elementary unit mesh, where the junction region is visible

Step 3: Reproduction of the finite element model in a polar series

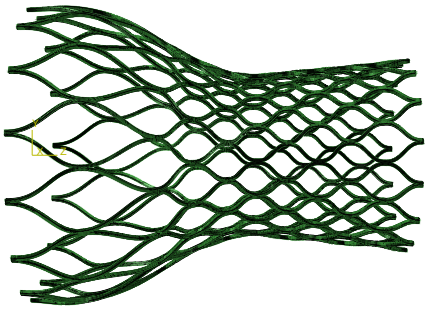
The information contained in the INP file is used as input for a Matlab algorithm (see B.1) that reproduces the mesh nodes and elements fifteen times in a polar series and computes the correspondent indexes. This code produces an INP file, that can be read in Abaqus, containing the newly elaborated mesh information about the complete structure of the prosthesis.

Step 4: Merging nodes of all junction regions

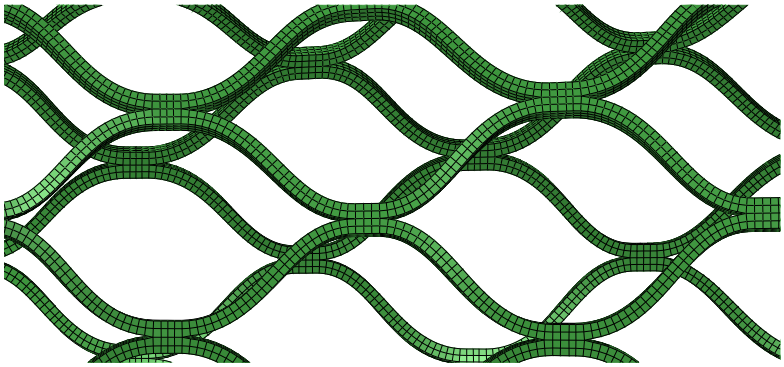
Finally the Abaqus program is used to repeat the merging operation, described at step 2, for all junction regions between the fifteen elementary units. So the final finite element model (see Figure 3.48) is ready for next step: the virtual simulation of the deployment.

3.3 Simulation of prosthesis implantation in native aortic root

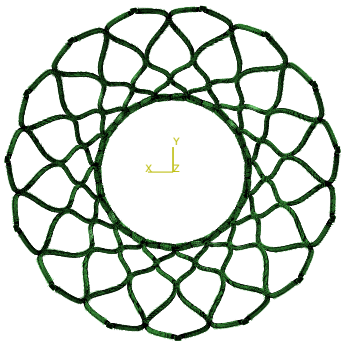
To perform a simulation of the prosthesis implantation in the native aortic root the Abaqus program is used. It is a software suite for finite element analysis and computer-aided engineering, that consists of five core software products, that used for performing simulations is Abaqus/CAE or "Complete Abaqus Environment". This is a software application used for the modeling and assembling (pre-processing) of mechanical components and analysis and visualization of the



(a)



(b)



(c)

Figure 3.48: (a) CoreValve mesh; (b) detail mesh; (c) superior view

obtained result. Abaqus/CAE is divided into functional units called modules. Each module contains only those tools that are relevant to a specific portion of the modeling task. The order of the modules in the menu corresponds to the logical sequence you follow to create a model. The modules are the following:

1. Part: create individual parts by sketching or importing their geometry.
2. Property: create material definitions and assign them to regions of parts.
3. Assembly: create and assemble part instances.
4. Step: create and define the analysis steps and associated analysis procedure.
5. Interaction: specify the interactions, such as contact, between regions of a model.
6. Load: specify loads, boundary conditions, and fields.
7. Mesh: create a finite element mesh.
8. Job: submit a job for analysis and monitor its progress.
9. Visualization: view analysis results.
10. Sketch: Create two-dimensional sketches.

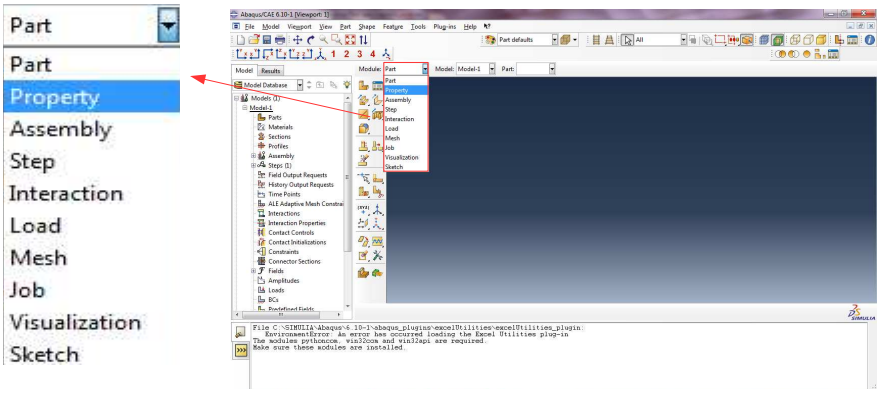


Figure 3.49: Abaqus interface

This modules are used to simulate the two main phases of stent crimping and stent deployment. The stent crimping consists in narrowing the CoreValve to introduce it into the delivery catheter. In reality this operation is performed manually by an expert operator, who uses some fresh water basins to gradually

cool the nitinol structure, thus the CoreValve becomes more flexible and deformable. To reproduce this operation virtually a cylinder has been introduced round the CoreValve for imposing a radial displacement that produces the structure shrinkage. This cylinder is also used in the phase of stent deployment for modeling the catheter, from which the CoreValve is released in the native valve of the patient. For simplicity, these two steps are performed one immediately after another and in both the aortic root is present. In the crimping phase the presence of the aortic root is ignored and the parts which interact are only the cylinder and the CoreValve. In the deployment phase the CoreValve interacts with the catheter and the aortic root to produce the final result. The steps performed to obtain the simulation are described as follows, with reference to most important Abaqus modules.

Part Three parts are introduced:

1. CoreValve: the model consists in the previously elaborated mesh (see section 3.2), the geometrical model is absent
2. cylinder: the geometrical model is created with Abaqus tools; the length is, 80 mm, and the diameter is 21 mm, 2 mm bigger than prosthesis diameter, which is 40 mm
3. aortic root: the model consists in the previously elaborated mesh (see section 3.1), the geometrical model is absent; the model represents the arterial wall with inner valve and calcium.

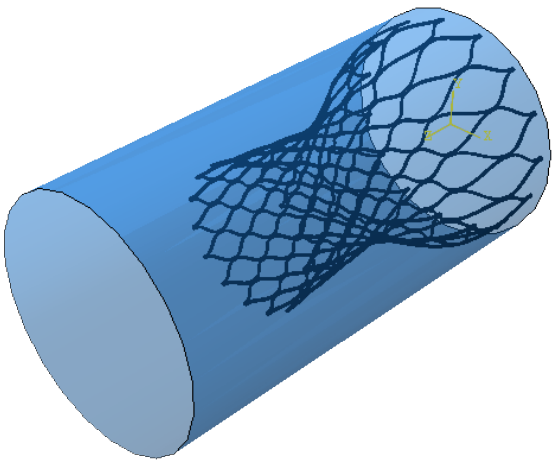


Figure 3.50: Initial configuration of the CoreValve and the cylinder

Property The material is assigned for each part:

1. CoreValve: the material used is nitinol; a pre-defined material model (present in Abaqus library) is used, which requires the insertion of fourteen constants [22] as parameters and the density, that is set at value of $6,5 * 10^{-9} Kgl^{-3}$
2. cylinder: the material is defined only by the density with the value of $6,7 * 10^{-9} Kgl^{-3}$
3. aortic root: the material used is linear elastic for all the components; for its definition the values set for each of the components is reported in Table 3.1.

	Arterial wall	Inner valve	Calcium
Density (Kgl^{-3})	$2 * 10^{-9}$	$1.1 * 10^{-9}$	$2 * 10^{-9}$
Young's Modulus (Pa)	2	1	12.6
Poisson's Ratio	0.45	0.45	0.3

Table 3.1: Inserted values for material definition

Step Two steps are created, for the two phases of crimping and deployment. The two steps follow a Dynamic Explicit analyis procedure. The Mass Scaling technique is used to accelerate the analysis, the time-step imposed is $1 * 10^{-6}$ sec for the first step and $1 * 10^{-7}$ sec for the second step.

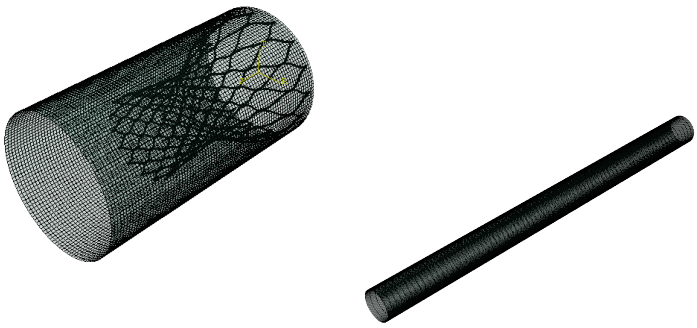
Interaction The type of contact between the two parts is of type tangential and normal. Moreover, it is imposed that the two parts cannot permeate themselves, and neither part can permeate itself. For the first step the contact is defined only between the CoreValve and the cylinder. For the second step the contact is defined only between CoreValve and aortic root and between catheter (cylinder) and CoreValve.

Load The load consists in boundary conditions. For the first step, a radial displacement toward the cylinder's central axis is imposed at cylinder surface, to allow for the shrinkage of the prosthesis until the diameter is 6 mm, that is the diameter value of a real catheter. For the second step a vertical upward translation of 90 mm is imposed on the cylinder, thus when the cylinder raises the CoreValve automatically enlarges itself.

Mesh The mesh of the CoreValve was prepared previously (see section 3.2.1). The mesh of the aortic root was prepared previously (see section 3.1.4). The cylinder mesh is defined with surface quadratic element.

Visualization The final analysis results have been visualized through the Abaqus interface, that permits viewing both the deformed configurations of the parts and a video of the simulations.

The initial and final phases of the crimping operation performed at the first step are shown in Figure 3.51, without the presence of the aortic root to better visualize the transformation. While Figure 3.52 shows four sequential phases of



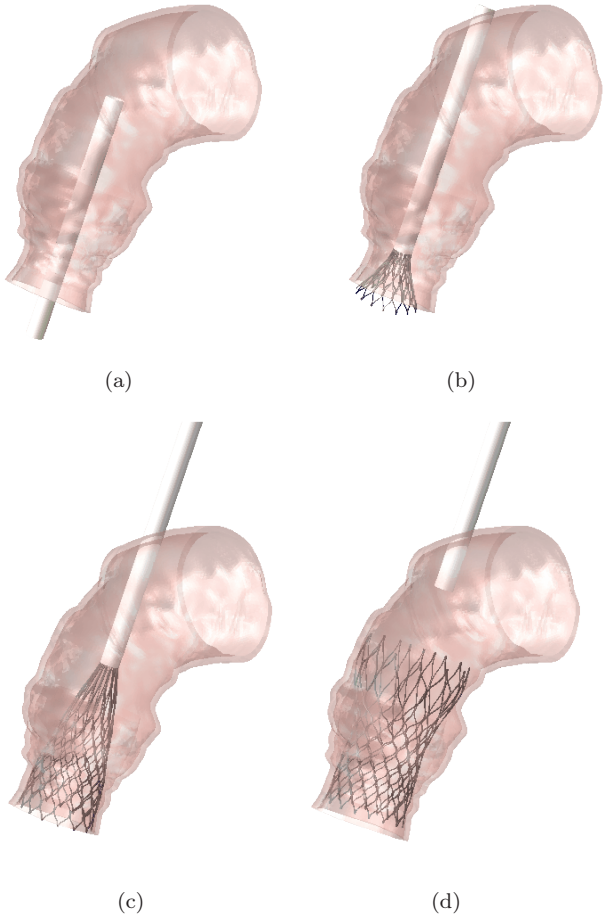
(a) (b)

Figure 3.51: Crimping: (a) initial phase, (b) final phase

the deployment of the prosthesis, performed at the second step. These results confirm that the developed models can be used to perform virtual simulations of transcatheter aortic valve implant. Moreover, the Abaqus software permits to analysis of the results, studying singly each deformed part, visualizing a map of the stress and the strain which act on the parts during transformation, etc. These tools are very useful in the phase of the TAVI planning, because the results of different configurations can be evaluated to found a better way for the implantation execution; for example the height of the CoreValve could be evaluated with respect to the arterial wall to ensure better positioning of the prosthesis. Moreover, computational analysis can be used as a tool to assess the feasibility and safety of TAVI in patients who are currently considered unsuitable for this procedure.

The important role of nitinol

The functioning of the CoreValve prosthesis is based on its material properties, nitinol, which are dependent on temperature. The difference in temperature between outside and inside of the human body make the deployment of the prosthesis in the native valve of the patient possible. When the prosthesis is



(a) (b)

(c) (d)

Figure 3.52: Deployment, second step: (a) the catheter is inserted, (b) the catheter begins to rise, (c) the catheter rises, (d) the CoreValve is released and fits arterial wall

cooled the nitinol becomes more flexible, thus the operator, using his hands, can modify its shape, shrinking the prosthesis for the insertion into the delivery catheter. Successively, when the prosthesis enters the body, the higher temperature allows it to recover the original shape, when it is released from catheter.

This material is a Shape Memory Alloys (SMA), that are a group of metallic materials which possess the ability to return to a previously defined shape when subjected to appropriate thermal procedure. The shape memory effect occurs due to a temperature and stress dependent shift in the material's crystalline structure between two different phases, martensite (low temperature phase) and austenite (high temperature phase). The temperature, where the phase transformation occurs, is called the transformation temperature. In the austenite

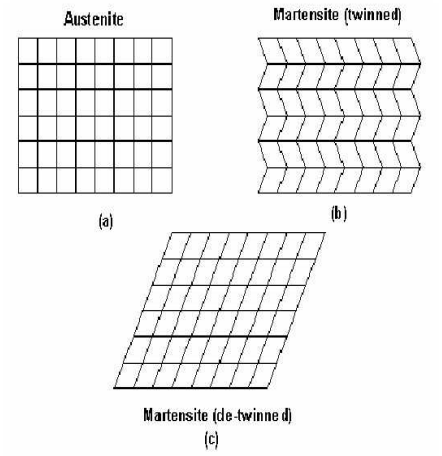


Figure 3.53: Crystalline arrangement of SMA in different phases

phase, the structure of the material is symmetrical; each "grain" of material is a cube with right angles (a). When the alloy cools, it forms the martensite phase and collapses to a structure with different shape (b). If an external stress is applied, the alloy will yield and deform to an alternate state (c). Now, if the alloy is heated again above the transformation temperature, the austenite phase will be formed and the structure of the material returns to the original "cubic" form (a). The shape memory effect must be "programmed" into the SMA alloys with an appropriate thermal procedure.

Chapter 4

Conclusions

The transcatheter aortic valve implant (TAVI) is a minimally invasive surgical procedure, established in recent years, that is performed to treat a set of patients, suffering from aortic valve stenosis and defined at high surgical risk for the traditional treatment of surgical valve replacement. The TAVI procedure consists in the implantation of a valvular prosthesis in the native aortic root of the patient, through the utilization of a delivery catheter introduced through vascular access. Currently the most used prostheses are the balloon-expendable Edward SAPIEN and the self-expandable CoreValve. Crucial aspects, on which the patient's outcome depends, are the positioning of the transcatheter aortic prosthesis as well as choosing the optimal device in terms of type and size; in this context the intervention planning procedure plays a determinant role for the post-operative performance of the prosthesis.

Recently, the finite element analysis (FEA) approach has been applied in several studies to investigate TAVI; in particular Auricchio et al. [6] have presented a patient-specific computational strategy which offers a useful tool to evaluate a balloon-expandable valve implant aiming at anticipating surgical operation outcomes. This thesis work, starting from this strategy, develops a new procedure for the pre-processing of the clinical data for performing TAVI simulation, through FEA. The goal of this procedure is to automate and accelerate the construction strategy of the aortic model and to increase the clinical cases on which the simulation strategy can be applied. The work mainly focuses on the development of a procedure for the creation of a patient-specific aortic model and the construction of a CoreValve device model, from clinical data to finite element models. The creation of a self-expandable prosthesis model enables the performance of virtual simulations on a greater number of patients. The entire procedure to develop the patient-specific model requires the use of

four programs Osirix, Matlab, Rhinoceros and Abaqus. The new procedure of construction of the aortic model, thanks to the implementation of some new algorithms for the Rhinoceros and Matlab programs, permits:

1. to include of the aortic arc in the model, the upper region to the inner valve;
2. to automate a phase of data elaboration, the selection of nodes from STL mesh (previously performed manually on Abaqus program aided);
3. to make almost completely automatic the geometrical model construction phase;
4. to reduce the necessary time of application of the procedure.

Moreover, the entire procedure has been used to build the aortic models of two patients subjected to self-expandable valve implant, whose data was provided by the *Istituto Clinico Sant'Ambrogio*. Finally, the developed models have been tested by performing virtual finite element simulations, the model of one of the two studied patients has been used to perform a complete simulation of the CoreValve prosthesis implantation. Currently, the entire procedure is still only partially automated and new algorithms could be edited to facilitate the application on a specific clinical case, especially because the final aim of this strategy is to provide a useful support tool for the physician in the phase of planning of the TAVI. Moreover, it would be useful to develop the CoreValve model in the other three sizes, currently available on the market and the procedure developed in this thesis work could be used for the construction of the models. Finally, the adopted methodology on finite element analysis should be validated setting up a protocol for comparing the simulation results with in-vivo post-implant measurements, possibly for a large number of patients. Besides the intrinsic limitations related to the complex system under investigation, the results of this work represent a solid base for the virtual planning of TAVI procedure, aiming at providing a useful tool to achieve optimal positioning and outcomes.

Appendix A

MATLAB codes for aortic model

A.1 Code for STL data elaboration

A.1.1 Main function

```
clear all
close all
clc

%%% READ %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

stl    = input('Did you load the ROOT stl file? Y/N  ','s');
name_p = input('What is the patient name   ','s');

if stl == 'N'
  wkdir=cd;
  [fn,pathname]= uigetfile('*.stl','Pick root STL_data (*.stl)');
  cd(pathname)
  [data,elems,x,y,z]=MC_readstl(fn);
  cd(wkdir);
  np=length(x);
else
    [fn,pathname]= uigetfile('*.mat','Pick STL_data (*.mat)');
```

```
    cd(pathname)
    load(fn);
end
counter=1;
figure(counter)
counter=counter+1;
MC_draw_stl(elems,data)

%%% SAVE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
name_data = [name_p '_input_data'];
save(name_data,'data','x','y','z','np','elems');

%%leggi dati calcio
wkdir=cd;
[fn_ca,pathname]= uigetfile('*.stl','Pick Ca STL_data (*.stl)');
cd(pathname)
[data_ca,elems_ca,xca,yca,zca]=MC_readstl(fn_ca);
cd(wkdir);
np_ca=length(xca);
%%% TRASLAZIONE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[data,trasl] = trasla(data,x,y,np);
%%trasl calcio
xca=xca-trasl(1);
yca=yca-trasl(2);
zca=zca-trasl(3);
data_ca=[xca,yca,zca];
%%% ROTAZIONE Y %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[data,RR,phi] = ruota(data,np,2);
rot_y = phi;
%%%rotazione calcio
for j=1:np_ca;
    data_ca(j,:) = (RR * data_ca(j,:)')';
end
%%%ROTAZIONE X%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[data,RR,phi] = ruota(data,np,1);
rot_x = phi;
%%%rotazione calcio
for j=1:np_ca;
    data_ca(j,:) = (RR * data_ca(j,:)')';
end
figure(counter)
counter=counter+1;
plot3(data(:,1),data(:,2),data(:,3),'r*')
hold on
plot3(data_ca(:,1),data_ca(:,2),data_ca(:,3),'k*')
grid on

%%% PICK 1 FOR CENTERLINE %%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(counter)
counter=counter+1;
plot(data(:,1),data(:,3),'o')
xlabel('x')
ylabel('z')
grid on
hold on

n_pti_CL = 100; % punti centerline


[pti,pol_z,counter] = centerline(1,n_pti_CL,counter);
pol_z1 = pol_z;
pti_x  = pti;

%estendiamo a 3D
cl_pt = [pti_x',zeros(n_pti_CL,1),pol_z1'];


%%% PICK 2 FOR CENTERLINE %%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
figure(counter)
counter=counter+1;
plot(data(:,2),data(:,3),'o')
xlabel('y')
ylabel('z')
grid on
hold on


[pti,pol_z,counter] = centerline(2,n_pti_CL,counter);
pol_z2 = pol_z;
pti_y  = pti;


% interpolo per determinare la y per i valori di z1 (pol_z1)
pti_z  = pol_z1;
%interploazione con spline cubica valutata in pti
pol_y = interp1(pol_z2,pti_y,pti_z,'spline');

CL = [pti_x', pol_y', pti_z'];


figure(counter)
counter=counter+1;
plot3(data(:,1),data(:,2),data(:,3),'ro')
xlabel('x')
ylabel('y')
zlabel('z')
grid on
hold on
plot3(CL(:,1),CL(:,2),CL(:,3))

%%% SAVE ROTATIONS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
name_data = [name_p '_data_save1'];
save(name_data,'data','x','y','z','np','elems','trasl',...
'rot_y','rot_x','CL');
```

```
l  = line_length(CL);
dl = 2.5;

n   = round(l/dl);
nTh =36;   %how many point along circumferencial direction
nz  = n;   %how many point along z-axis

figure(counter)
counter=counter+1;
plot3(data(:,1),data(:,2),data(:,3),'ro')
xlabel('x')
ylabel('y')
zlabel('z')
grid on
hold on
plot3(CL(:,1),CL(:,2),CL(:,3))


%%% ORDINA e SELEZIONA PUNTI DI OGNI SEZIONE %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[Xin,Yin,Zin,counter,thick] = sel_prova(nz,nTh,CL,data,elems,counter);

name_file = [name_p '_Rhinoscript.txt'];
name_ca=[name_p '_calcio.txt'];

nTh = nTh+1;

figure(counter)
counter=counter+1;
mesh(Xin_new,Yin_new,Zin_new)
xlabel('x')
ylabel('y')
zlabel('z')

%%%%%%%%%%%% SCRIPT per Rhino %%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
directory = ['/Users/Giovanni/Documents/MATLAB/Matlab_repository/' name_p '/'];

cd (directory)
write_script_rhino3(nz,nTh,Xin_new,Yin_new,Zin_new, name_file,CL,thick);
write_script_calcio(name_ca,data_ca,elems_ca,np_ca);
```

A.1.2 Centerline

```
function [pti pol_z counter] = centerline(flag,n_pti_CL,counter)

display('Pick 5 points on the pseudo-centerline [first-bottom, last-top]')
p1 = ginput(5);

if flag==1
dx=p1(1,2)+3;
dz=p1(1,1);
p_xy=p1(:,2)-dx;
p_z=p1(:,1)-dz;


fit_fn=fit(p_xy,p_z,'exp1');
var_xy=fit_fn(p_xy);
z_fit=p_xy;

var_xy=var_xy+dz;
z_fit=z_fit+dx;

else
var_xy=p1(:,1);
z_fit=p1(:,2);

end
pol_z  = linspace(min(p1(:,2)),max(p1(:,2)),n_pti_CL);
%interploazione valutata in pti
pti = interp1(z_fit,var_xy,pol_z,'spline');

figure(counter)
plot(var_xy,z_fit,'b')
```

```
hold on
plot(var_xy,z_fit,'bo')
hold on
plot(p1(:,1),p1(:,2),'r')
hold on
plot(pti,pol_z,'g*')
grid on
counter=counter+1;
```

A.1.3 Nodes elaboration

```
function [Xin,Yin,Zin,counter,thick] = sel_prova(nz,nTh,CL,data,elems,counter)

sections = [];
Thf = 2*pi;
dTh = Thf/nTh;
par_diff=4;

Th_int = zeros(nTh+1,1);

[CL2 tan counter]=vettori_tangenti(CL,nz,counter);
CL=[];
CL=CL2';

N = nz*nTh; % numero totale di punti
%initialize
Xin = zeros(nz,nTh);
Yin = zeros(nz,nTh);
Zin = zeros(nz,nTh);



P_cart_old = zeros(nTh,3);

thick = input('Thickness of the arterial wall [mm] =    ');

hw = waitbar(0,'Computing in progress. Please wait...');
dist_med=[];
for i=1:nz; %2: buttiamo via la primissima sezione (spesso incompleta)
```

```
waitbar(i/(nz),hw);

a = 0; %indice di sezione
P_cart = zeros(nTh,3);
ind_j=[];
p_j = CL(i,:); %point
e_n=tan(:,i);
x=[];
y=[];
z=[];


[sec_tmp,id_elem] = MC_cut_stl_basic(data,elems,e_n,CL(i,:));
s = size(sec_tmp,1);

%calcolo il baricentro e traslo sull'origine (0,0,0)
sec_tmp_tras = trasla_bar(sec_tmp,p_j);
x = sec_tmp_tras(:,1);
y = sec_tmp_tras(:,2);
z = sec_tmp_tras(:,3);

mat=rot_piano(e_n,x,y,z,1);
x=mat(1,:)';
y=mat(2,:)';
z=mat(3,:)';

nodes_p      = zeros(s,3);
[Th, rho, z] = cart2pol(x,y,z);

% [0 < Theta < 2pi], not [-pi < Theta < pi]
Th = Th + pi;
nodes_p = [Th, rho, z];
enter = 0;
ind_j=0;
for j =2:nTh+1;
  Th_int(j) = Th_int(j-1) + dTh;
  points_ok = find(nodes_p(:,1) > Th_int(j-1) &
```

```
nodes_p(:,1) < Th_int(j));

if points_ok ~= 0

    x_pol=nodes_p(points_ok,1);
    y_pol=nodes_p(points_ok,2);
    z_pol=nodes_p(points_ok,3);
    [x1, y1, z1] = pol2cart(x_pol,y_pol,z_pol);
    x1=-x1;
    y1=-y1;

    distan=zeros(1,length(points_ok));
    c_points_ok  = [x1, y1, z1];

    for k=1:size(c_points_ok,1)

        distan(k)=dist([0 0 0],(c_points_ok(k,:))');
    end

    [distan,ind]=sort(distan);

    for h=1:size(c_points_ok,1)
        c_points_ok_new(h,:)=c_points_ok(ind(h),:);
    end

    c_points_ok=c_points_ok_new;

    if length(distan)>1
        distan_new=calcola_distanze(distan);
        c_points_ok(find(distan_new==0),:)=[];
        distan_new(find(distan_new==0))=[];
        q=quantile(distan_new,0.9);
        c_points_ok(find(distan_new<q),:)=[];
        distan_new(find(distan_new<q))=[];
        dist_med(j)=mean(distan_new);
        if j>2
            if dist_med(j-1)~=0
            diff_m=abs(dist_med(j)-dist_med(j-1));
```

```
                else
                diff_m=par_diff;
                end
                if diff_m>par_diff
                    enter=1;
                    dist_med(j)=dist_med(j-1);
                    ind_j=[ind_j j-1];
                    c_points_ok_new=[];
                    points_ok=[];
                    c_points_ok=[];
                    distan=[];
                    distan_new=[];
                    x_pol=[];
                    y_pol=[];
                    z_pol=[];
                    x1=[];
                    y1=[];
                    z1=[];
                    continue
                end
            end
            P_cart(j-1,:) = [mean(c_points_ok(:,1)),mean(c_points_ok(:,2)),...
            mean(c_points_ok(:,3))];
        else
            dist_med(j)=distan;
            P_cart(j-1,:) = [c_points_ok(:,1),c_points_ok(:,2),c_points_ok(:,3)];
        end
if i==100
          figure(j+15)
          plot3(P_cart(j-1,1),P_cart(j-1,2),P_cart(j-1,3),'ro')
          hold on
          plot3(x1, y1, z1,'b.')
          hold on
          plot3(0,0,0,'k*')
          hold on
          plot3(x,y,z,'g.')
          grid on
end
```

```
    else
         enter=1;
         ind_j=[ind_j j-1];
         if j>2
         dist_med(j)=dist_med(j-1);
         else
         dist_med(j)=0;
         end

    end

    c_points_ok_new=[];
    points_ok=[];
    c_points_ok=[];
    distan=[];
    distan_new=[];
    x_pol=[];
    y_pol=[];
    z_pol=[];
    x1=[];
    y1=[];
    z1=[];
  end

  if enter == 1

    %%% Filling of the missing slices with new nodes%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    ind_j(1)=[];
    for p=1:length(ind_j)
        if ind_j(p)==1
            t=2:nTh;
            l=sort(t,2,'descend');
        end
        if ind_j(p)==nTh
            t=1:nTh-1;
            l=sort(t,2,'descend');
        end
```

```
        if ind_j(p)>1 && ind_j(p)<nTh
        a=1:(ind_j(p)-1);
        a_sor=sort(a,2,'descend');
        b=(ind_j(p)+1):nTh;
        b_sor=sort(b,2,'descend');
        t=[b a];
        l=[a_sor b_sor];
        end
        for f=1:nTh-1;
        if P_cart(l(f),1)~=0
            zeta1=P_cart(l(f),3);
            rag1=dist([0 0 0],(P_cart(l(f),:))');
            break
        end
        end
        for f=1:nTh-1;
            if P_cart(t(f),1)~=0
                 zeta=P_cart(t(f),3);
                 rag=dist([0 0 0],(P_cart(t(f),:))');
                 break
            end
        end
        [xj,yj,zj]=pol2cart(((ind_j(p)-1)*dTh)+dTh/2,mean([rag1
        rag]),mean([zeta1 zeta]));
        P_cart(ind_j(p),:)=[-xj,-yj,zj];
    end
    enter=0;
    ind_j=[];
end
mat=rot_piano(e_n,x,y,z,2);
x=mat(1,:)';
y=mat(2,:)';
z=mat(3,:)';
mat=rot_piano(e_n,P_cart(:,1),P_cart(:,2),P_cart(:,3),2);
P_cart(:,1)=mat(1,:)';
P_cart(:,2)=mat(2,:)';
P_cart(:,3)=mat(3,:)';
P_cart = trasla_bar(P_cart,-p_j);
```

```
  figure(i+60)
  plot3(P_cart(:,1),P_cart(:,2),P_cart(:,3),'ro')
  hold on
  plot3(p_j(1),p_j(2),p_j(3),'k*')
  grid on
  plot3(x'+p_j(1),y'+p_j(2),z'+p_j(3),'b.')

  % memorizzo P_pol
  P_cart_old = P_cart;

  Xin(i,:) = P_cart(:,1)';
  Yin(i,:) = P_cart(:,2)';
  Zin(i,:) = P_cart(:,3)';

 sections = [sections; sec_tmp];

end
figure(counter)
counter=counter+1;
str='ro';
plot_sections(sections,str)
grid on
```

Computation of centerline tangent vector

```
function [cv  cdv counter]=vettori_tangenti(CL,nz,counter)

curve=cscvn(CL');

figure(20)
plot3(CL(:,1),CL(:,2),CL(:,3),'r')
xlabel('x')
ylabel('y')
zlabel('z')
grid on
hold on
fnplt(curve);
```

```
b=curve.breaks;
t=linspace(ceil(min(b)),floor(max(b)),nz);
cv = fnval(curve, t);
der=fnder(curve);
cdv = fnval(der, t);

figure(counter)
plot3(cv(1,:),cv(2,:),cv(3,:),'r.');
hold on
quiver3(cv(1,:),cv(2,:),cv(3,:), cdv(1,:),cdv(2,:),cdv(3,:),0.5);
hold on
fnplt(curve);
grid on
counter=counter+1;
```

Computation of the distances

```
function distan_new=calcola_distanze(distan)
      %parametro per l'esclusione dei punti interni
      par_dist=2;
      len_dist=length(distan);
      distan_new(len_dist)=distan(len_dist);
      for h=1:len_dist-1;
          diff=abs(distan(len_dist-(h-1))-distan(len_dist-(h)));
          if diff<par_dist
              distan_new(len_dist-(h))=distan(len_dist-(h));h
          else
              distan_new(1:(len_dist-(h)))=0;
              break
          end
      end
```

Rotation of the plane of transversal section

```
function mat=rot_piano(e_n,x,y,z,i);
r=vrrotvec([0 0 1],e_n);
if i==1
r(4)=-r(4);
end
m = vrrotvec2mat(r);
```

```
mat=m*[x,y,z]';
```

A.1.4 Editing RVB file for model construction

```
function write_script_rhino3(nz,nTh,Xin,Yin,Zin,...
name_file,CL,thick)

fid = fopen(name_file,'wt');

%%%% IN
fprintf(fid,'%s\n', 'Option Explicit');
fprintf(fid,'%s\n', 'Call Main()');
fprintf(fid,'%s\n', 'Sub Main()');
fprintf(fid,'%s\n', 'Rhino.RenderSettings 1');
fprintf(fid,'%s %d %s\n','Dim arrPointCloud(',nTh-1,'),...
blnCompare');

%%%%%%%%%%%% superficie interna %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
i=[];
j=[];
for i=1:nz
    %prima curva
    fprintf(fid,'%s%d\n','Dim curve_i',i);
    for j=1:nTh-1
         fprintf(fid,'%s %d %s %d %s %d %s %d
         %s\n','arrPointCloud(',j-1,') = Array(',Xin(i,j),',',
         Yin(i,j),',' ,Zin(i,j),')');
    end
    fprintf(fid,'%s %d %s %d %s %d %s %d %s\n',...
    'arrPointCloud(',j,') = Array(', Xin(i,j+1), ',...
     ', Yin(i,j+1), ', ', Zin(i,j+1),')');
fprintf(fid,'%s\n','If IsArray(arrPointCloud) Then');
    fprintf(fid,'%s%d %s\n','curve_i',i,'=Rhino.AddCurve    (arrPointCloud)');
fprintf(fid,'%s\n', 'End If');
    if i>1
    fprintf(fid,'%s%d %s%d %s\n','blnCompare =
    Rhino.CurveDirectionsMatch(curve_i',i,', curve_i',i-1,')');

fprintf(fid,'%s\n','If blnCompare = False Then');
```

```
fprintf(fid,'%s%d\n','Rhino.ReverseCurve curve_i',i);

fprintf(fid,'%s\n','End If');
    end
end
    fprintf(fid,'%s\n','Dim curve_tot,sup_int');


    fprintf(fid,'%s','curve_tot = Array(');
    for i=1:nz
        if i==nz
        fprintf(fid,'%s%d %s','curve_i',i);
        else
        fprintf(fid,'%s%d %s','curve_i',i,',');
        end
    end
    fprintf(fid,'%s\n',')');
    fprintf(fid,'%s\n','sup_int=Rhino.AddLoftSrf (curve_tot)');
    fprintf(fid,'%s\n','Dim curve_del');
fprintf(fid,'%s','curve_del = Array(');
    for i=2:nz-1
        if i==nz-1
        fprintf(fid,'%s%d %s','curve_i',i);
        else
        fprintf(fid,'%s%d %s','curve_i',i,',');
        end
    end
     fprintf(fid,'%s\n',')');
fprintf(fid,'%s\n','rhino.DeleteObjects curve_del');

    %%%%%%%%%%9 punti per i foglietti%%%%%%%%%%%%%%%%%
    fprintf(fid,'%s\n','Dim
    p1,p2,p3,p4,p5,p6,p7,p8,p9,pc1,pc2,pc3,pc4,pc5,...
    pc6,pc7,pc8,pc9');
    for i=1:9
fprintf(fid,'%s%d %s %d %s\n','p',i,' =
Rhino.GetPointOnSurface(sup_int(0), "Point ',i,...
```

```
' of leaflets")');
fprintf(fid,'%s%d %s%d %s \n',' pc',i,'=Rhino.AddPoint
(p',i,')');
    end

    fprintf(fid,'%s\n','Dim plane1,plane2,plane3,plane4,p0,plane5,plane6,plane7,cir');
    fprintf(fid,'%s\n','cir=Rhino.AddCircle3Pt(p1, p8, p5)');
fprintf(fid,'%s\n','p0 = Rhino.CircleCenterPoint(cir)');

    %%%%%%%%%piani di taglio%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fprintf(fid,'%s\n','plane1 = Rhino.AddCutPlane(sup_int(0),...
     p1, p5, Rhino.VectorCreate(p5, p8))');
    fprintf(fid,'%s\n','plane2 =Rhino.AddCutPlane(sup_int(0),...
    p1,p2, Rhino.VectorCreate(p2,p0))');
    fprintf(fid,'%s\n','plane3 = Rhino.AddCutPlane(sup_int(0),...
    p4,p5, Rhino.VectorCreate(p4,p0))');
    fprintf(fid,'%s\n','plane4 = Rhino.AddCutPlane(sup_int(0),...
    p7,p8, Rhino.VectorCreate(p8,p0))');
    fprintf(fid,'%s\n','plane5 = Rhino.AddCutPlane(sup_int(0),...
    p2,p3, Rhino.VectorCreate(p3,p4))');
    fprintf(fid,'%s\n','plane6 = Rhino.AddCutPlane(sup_int(0),...
    p7,p4, Rhino.VectorCreate(p6,p7))');
    fprintf(fid,'%s\n','plane7 = Rhino.AddCutPlane(sup_int(0),...
    p7,p9, Rhino.VectorCreate(p9,p2))');
    fprintf(fid,'%s\n','Dim
    int1,int2,int3,int4,int5,int6,int7,plane_del');
    for i=2:7
fprintf(fid,'%s%d %s%d %s\n','int',i,' =
Rhino.SurfaceSurfaceIntersection(sup_int(0), plane',i,',,
True)');
    end
fprintf(fid,'%s\n','plane_del = Array(plane2, plane3, plane4,
plane5, plane6, plane7,cir)');
fprintf(fid,'%s\n','rhino.DeleteObjects plane_del');
    fprintf(fid,'%s\n','Dim inter1,inter2,inter3,inter4,inter5,inter6,del');
fprintf(fid,'%s\n','inter1 = find_curve(int2, p1, p2,...
 null)');
fprintf(fid,'%s\n','inter2 = find_curve(int3, p4, p5,...
```

```
null)');
    fprintf(fid,'%s\n','inter3 = find_curve(int4, p7, p8,...
    null)');
    fprintf(fid,'%s\n','inter4 = find_curve(int5, p4, p2,p3)');
    fprintf(fid,'%s\n','inter5 = find_curve(int6, p4, p7,p6)');
    fprintf(fid,'%s\n','inter6 = find_curve(int7, p7, p2,p9)');
    fprintf(fid,'%s\n','Dim split1,area1,area2,split_inf,...
    split_sup');
fprintf(fid,'%s\n','split1 = rhino.SplitBrep(sup_int(0),... plane1, True)');
   fprintf(fid,'%s\n',' rhino.deleteObject plane1');
fprintf(fid,'%s\n','area1 = rhino.SurfaceArea(split1(0))');
fprintf(fid,'%s\n','area2 = rhino.SurfaceArea(split1(1))');

fprintf(fid,'%s\n','If  area1(0) < area2(0) Then');
fprintf(fid,'%s\n','split_inf = split1(0)');
fprintf(fid,'%s\n','split_sup = split1(1)');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','split_inf = split1(1)');
fprintf(fid,'%s\n','split_sup = split1(0)');
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','Dim sCrvs,taglio');
fprintf(fid,'%s\n','taglio = rhino.JoinCurves(Array(inter4,
inter5, inter6))');
fprintf(fid,'%s\n','sCrvs = " _SelID " & taglio(0)');
fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
fprintf(fid,'%s\n','rhino.SelectObject split_inf');
fprintf(fid,'%s\n','Rhino.Command "_Split"& SCrvs &" _Enter",
False');
fprintf(fid,'%s\n','rhino.DeleteObjects taglio(0)');

fprintf(fid,'%s\n','Dim  last,s_infinf,s_infsup');
fprintf(fid,'%s\n','last = Rhino.LastCreatedObjects() ');
fprintf(fid,'%s\n','If rhino.IsPointOnSurface(last(0), p1, 0.1)Then');
fprintf(fid,'%s\n','s_infsup = last(0)');
fprintf(fid,'%s\n','s_infinf = last(1)');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','s_infsup = last(1)');
fprintf(fid,'%s\n','s_infinf = last(0)');
```

```
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','sCrvs = Null');
fprintf(fid,'%s\n','Dim Crvs,crv');
fprintf(fid,'%s\n','Crvs = Array(inter1, inter2)');
fprintf(fid,'%s\n','rhino.SelectObjects Crvs');
    fprintf(fid,'%s\n','For Each crv In Crvs');
fprintf(fid,'%s\n','sCrvs = sCrvs & " _SelID " & crv');
    fprintf(fid,'%s\n','Next');
fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
fprintf(fid,'%s\n','rhino.SelectObject s_infsup');
fprintf(fid,'%s\n','Rhino.Command "_Split"& SCrvs &" _Enter",
 False');
fprintf(fid,'%s\n','last = Rhino.LastCreatedObjects()');
fprintf(fid,'%s\n','Dim i,joi1,joi2,joi3,s_ok');
fprintf(fid,'%s\n','joi1 = Null');
fprintf(fid,'%s\n','For i=0 To UBound(last)');
fprintf(fid,'%s\n','If (rhino.IsPointOnSurface(last(i),...
 p1, 0.1) And rhino.IsPointOnSurface(last(i), p4, 0.1)) Then');
fprintf(fid,'%s\n','joi3 = last(i)');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','If isnull(joi1) Then');
fprintf(fid,'%s\n','joi1 = last(i)');
fprintf(fid,'%s\n','Else ');
fprintf(fid,'%s\n','joi2 = rhino.JoinSurfaces(Array(joi1, last(i)), True)');
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','Next');
fprintf(fid,'%s\n','last = Null');
    fprintf(fid,'%s\n','area1 = Null');
fprintf(fid,'%s\n','area2 = Null');
fprintf(fid,'%s\n','last = Array(joi2, joi3)');
fprintf(fid,'%s\n','area1 = rhino.SurfaceArea(last(0))');
fprintf(fid,'%s\n','area2 = rhino.SurfaceArea(last(1))');

fprintf(fid,'%s\n','If  area1(0) < area2(0) Then');
fprintf(fid,'%s\n','s_ok = last(1)');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','s_ok = last(0)');
```

```
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','sCrvs = Null');
fprintf(fid,'%s\n','sCrvs = " _SelID " & inter3');
fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
fprintf(fid,'%s\n','rhino.SelectObject s_ok');
fprintf(fid,'%s\n','Rhino.Command "_Split"& SCrvs &" _Enter",
False');
fprintf(fid,'%s\n','rhino.UnselectAllObjects()');
    fprintf(fid,'%s\n','Dim arco1,arco2,arco3');
fprintf(fid,'%s\n','rhino.SelectObject pc1');
fprintf(fid,'%s\n','rhino.SelectObject pc5');
fprintf(fid,'%s\n','arco1 = build_arc(p1, p5)');
fprintf(fid,'%s\n','rhino.SelectObject pc5');
fprintf(fid,'%s\n','rhino.SelectObject pc8');
fprintf(fid,'%s\n','arco2 = build_arc(p5, p8)');
fprintf(fid,'%s\n','rhino.SelectObject pc8');
fprintf(fid,'%s\n','rhino.SelectObject pc1');
fprintf(fid,'%s\n','arco3 = build_arc(p8, p1)');
   fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
fprintf(fid,'%s\n','rhino.SelectObject arco1');
fprintf(fid,'%s\n','rhino.SelectObject inter1');
fprintf(fid,'%s\n','rhino.SelectObject inter4');
fprintf(fid,'%s\n','rhino.SelectObject inter2');
fprintf(fid,'%s\n','Rhino.Command "_NetworkSrf" & " _Enter",
False');
fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
fprintf(fid,'%s\n','rhino.SelectObject arco2');
    fprintf(fid,'%s\n','rhino.SelectObject inter2');
fprintf(fid,'%s\n','rhino.SelectObject inter5');
fprintf(fid,'%s\n','rhino.SelectObject inter3');
fprintf(fid,'%s\n','Rhino.Command "_NetworkSrf" & " _Enter",
False');
    fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
fprintf(fid,'%s\n','rhino.SelectObject arco3');
fprintf(fid,'%s\n','rhino.SelectObject inter3');
fprintf(fid,'%s\n','rhino.SelectObject inter6');
fprintf(fid,'%s\n','rhino.SelectObject inter1');
fprintf(fid,'%s\n','Rhino.Command "_NetworkSrf" & " _Enter",
```

```
    False');
fprintf(fid,'%s\n','Rhino.UnselectAllObjects()');
    fprintf(fid,'%s\n','del=Array(pc1,pc2,pc3,pc4,pc5,...
    pc6,pc7,pc8,pc9,arco1,arco2,arco3,inter1,inter2,...
    inter3,inter4,inter5,inter6)');
    fprintf(fid,'%s\n','rhino.deleteObjects del');
    %%%%%%%%%%%% superficie esterna %%%%%%%%%%%%%%%%
   fprintf(fid,'%s\n',' Dim vect,vect1,vect2,punto');
for i=1:nz
    %prima curva
    fprintf(fid,'%s%d\n','Dim curve',i);
    for j=1:nTh-1
         fprintf(fid,'%s %d %s %d %s %d %s %d %s %d %s %d
         %s\n','vect=rhino.vectorCreate(Array(',CL(i,1),',...
         ',CL(i,2),',',CL(i,3),'),Array(',Xin(i,j),',',...
          Yin(i,j),',' ,Zin(i,j),'))');
         fprintf(fid,'%s %d %s\n',' vect1=Rhino.VectorDivide(vect,
         (rhino.VectorLength(vect) /
         (rhino.VectorLength(vect)+',thick,'))))');
         fprintf(fid,'%s\n','vect2=Rhino.VectorReverse (vect1)');
         fprintf(fid,'%s %d %s %d %s %d %s\n',...
         'punto=Rhino.PointAdd(Array(',CL(i,1),',...
         ',CL(i,2),',',CL(i,3),'), vect2)');
         fprintf(fid,'%s %d %s\n',...
         'arrPointCloud(',j-1,') = punto');
         if j==1
           fprintf(fid,'%s %d %s\n','arrPointCloud(',nTh-1,')=
           punto');
         end
    end

fprintf(fid,'%s\n','If IsArray(arrPointCloud) Then');
    fprintf(fid,'%s%d %s\n','curve',i,'=Rhino.AddCurve
    (arrPointCloud)');
fprintf(fid,'%s\n', 'End If');
    if i>1
    fprintf(fid,'%s%d %s%d %s\n','blnCompare =
    Rhino.CurveDirectionsMatch(curve',i,', curve',i-1,')');
```

```
fprintf(fid,'%s\n','If blnCompare = False Then');

fprintf(fid,'%s%d\n','Rhino.ReverseCurve curve',i);

fprintf(fid,'%s\n','End If');
    end
end
    fprintf(fid,'%s\n','Dim curve,sup_est');


    fprintf(fid,'%s','curve = Array(');
    for i=1:nz
        if i==nz
        fprintf(fid,'%s%d %s','curve',i);
        else
        fprintf(fid,'%s%d %s','curve',i,',');
        end
    end
    fprintf(fid,'%s\n',')');
    fprintf(fid,'%s\n','sup_est=Rhino.AddLoftSrf (curve)');

    %%%%%%%%%%%% superfici superiore e inferiore%%%%%%%%%%%%%%
    fprintf(fid,'%s\n','Dim sup');
    fprintf(fid,'%s%d %s%d %s\n','blnCompare =
    Rhino.CurveDirectionsMatch(curve_i',i,', curve',i,')');

fprintf(fid,'%s\n','If blnCompare = False Then');

fprintf(fid,'%s%d\n','Rhino.ReverseCurve curve_i',i);

fprintf(fid,'%s\n','End If');
    fprintf(fid,'%s%d %s%d %s\n','sup=Rhino.AddLoftSrf
    (Array(curve',i,', curve_i',i,'))');
    fprintf(fid,'%s\n','Dim inf');
    fprintf(fid,'%s%d %s%d %s\n','blnCompare =
    Rhino.CurveDirectionsMatch(curve_i',1,', curve',1,')');
```

```
fprintf(fid,'%s\n','If blnCompare = False Then');

fprintf(fid,'%s%d\n','Rhino.ReverseCurve curve_i',1);

fprintf(fid,'%s\n','End If');
    fprintf(fid,'%s\n','inf = Rhino.AddLoftSrf(Array(curve1, curve_i1))');
    fprintf(fid,'%s\n','rhino.JoinSurfaces Array(s_infinf, inf(0),
    sup_est(0), sup(0), split_sup), True');
    fprintf(fid,'%s\n','Rhino.DeleteObjects curve');
    fprintf(fid,'%s\n','Rhino.DeleteObjects curve_tot');

    fprintf(fid,'%s\n','End Sub');

    fprintf(fid,'%s\n','Function find_curve(int, p1, p2,p3)');
    fprintf(fid,'%s\n','Dim text,curve,i,j,arr');
fprintf(fid,'%s\n','text = Array(int(0, 1))');
fprintf(fid,'%s\n','For j = 1 To UBound(int)');
fprintf(fid,'%s\n','arr = Array(int(j, 1))');
fprintf(fid,'%s\n','text = rhino.JoinArrays(arr, text)');
fprintf(fid,'%s\n','Next');
fprintf(fid,'%s\n','curve = rhino.JoinCurves(text, True)');

fprintf(fid,'%s\n','Dim t0,t1,inter');
fprintf(fid,'%s\n','For i = 0 To UBound(curve)');
    fprintf(fid,'%s\n','If rhino.IsCurveClosed(curve(i)) Then');
fprintf(fid,'%s\n','t0 = Rhino.CurveClosestPoint(curve(i), p1)');
fprintf(fid,'%s\n','t1 = Rhino.CurveClosestPoint(curve(i),
    p2)');
fprintf(fid,'%s\n','inter = Rhino.AddSubCrv(curve(i), t0,
t1)');
fprintf(fid,'%s\n','If Rhino.IsPointOnCurve(inter, p3,,
    0.1)  Then');
fprintf(fid,'%s\n','Exit For');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','rhino.DeleteObject inter');
fprintf(fid,'%s\n','t0 =
    Rhino.CurveClosestPoint(curve(i), p2)');
fprintf(fid,'%s\n','t1 =
```

```
    Rhino.CurveClosestPoint(curve(i), p1)');
fprintf(fid,'%s\n','inter = Rhino.AddSubCrv(curve(i),
t0, t1)');
fprintf(fid,'%s\n','Exit For');
fprintf(fid,'%s\n','End If');
        fprintf(fid,'%s\n',' Else');
        fprintf(fid,'%s\n',' If Rhino.IsPointOnCurve(curve(i), p1,,
        0.1) Then');
fprintf(fid,'%s\n','t0 = Rhino.CurveClosestPoint(curve(i),
    p1)');

fprintf(fid,'%s\n','t1 = Rhino.CurveClosestPoint(curve(i),
    p2)');

fprintf(fid,'%s\n','inter = Rhino.AddSubCrv(curve(i), t0, t1)');

fprintf(fid,'%s\n','If isnull(inter)   Then');
fprintf(fid,'%s\n','t0 = Rhino.CurveClosestPoin(curve(i)
, p2)');

fprintf(fid,'%s\n','t1 = Rhino.CurveClosestPoint(curve(i),
    p1)');

fprintf(fid,'%s\n','inter = Rhino.AddSubCrv(curve(i), t0,
    t1)');

fprintf(fid,'%s\n','If isnull(inter) = False Then');
fprintf(fid,'%s\n','Exit For');
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','Exit For ');
fprintf(fid,'%s\n','End If');
        fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','End If');

fprintf(fid,'%s\n','Next');

    fprintf(fid,'%s\n','text = Array(curve(0))');
```

```
fprintf(fid,'%s\n','For j = 1 To UBound(curve)');
fprintf(fid,'%s\n','arr = Array(curve(j))');
fprintf(fid,'%s\n','text = rhino.JoinArrays(arr, text)');
fprintf(fid,'%s\n','Next');
fprintf(fid,'%s\n','rhino.DeleteObjects text');
fprintf(fid,'%s\n','find_curve = inter');
    fprintf(fid,'%s\n','End Function');
    fprintf(fid,'%s\n','Function build_arc(arrPt1, arrPt2)');
fprintf(fid,'%s\n','Dim  curve, mid, dist,rag,
 cat,vect,origin,vectr,ang,plane,arc,p,del');
fprintf(fid,'%s\n','Rhino.Print "Inserire il raggio:" ');
fprintf(fid,'%s\n','rag = Rhino.Getreal');
fprintf(fid,'%s\n','curve = rhino.AddCurve(Array(arrPt1,
    arrPt2))');
fprintf(fid,'%s\n','mid = rhino.CurveMidPoint(curve)');
fprintf(fid,'%s\n','dist = rhino.CurveLength(curve)');
fprintf(fid,'%s\n','If rag > (dist / 2) Then');
fprintf(fid,'%s\n','cat = Sqr((rag ^ 2) - ((dist / 2) ^
    2))');
fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','Rhino.MessageBox "Il valore
del raggio è troppo piccolo!"');
fprintf(fid,'%s\n','Rhino.Print "Inserire il raggio:" ');
fprintf(fid,'%s\n','rag = Rhino.Getreal');
fprintf(fid,'%s\n','cat = Sqr((rag ^ 2) - ((dist / 2) ^ 2))');
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','vect = rhino.VectorCreate(mid, arrPt1)');
fprintf(fid,'%s\n','vectr = Rhino.VectorRotate(vect, 90.0, Array(0, 0, 1))');
    fprintf(fid,'%s\n','vect = rhino.VectorDivide(vectr,
    (rhino.VectorLength(vectr) / cat))');
fprintf(fid,'%s\n','origin = rhino.PointAdd(mid, vect)');
    fprintf(fid,'%s\n','p = Rhino.AddPoint(origin)');
fprintf(fid,'%s\n','ang = rhino.angle2(Array(arrPt1, origin),
    Array(arrPt2, origin))');
    fprintf(fid,'%s\n','plane = rhino.PlaneFromPoints(origin,
    arrPt1, arrPt2)');
fprintf(fid,'%s\n','If ang(0) < ang(1) Then');
fprintf(fid,'%s\n','arc = rhino.AddArc(plane,...
```

```
 rag, ang(0))');
fprintf(fid,'%s\n','End If');
fprintf(fid,'%s\n','Dim startp,endp,dist1,dist2');
fprintf(fid,'%s\n','startp = rhino.CurveStartPoint(arc)');
fprintf(fid,'%s\n','endp = rhino.curveEndPoint(arc)');
fprintf(fid,'%s\n','dist1 = rhino.distance(startp, arrPt1)');
fprintf(fid,'%s\n','dist2 = rhino.distance(startp, arrPt2)');
fprintf(fid,'%s\n','If dist1 > dist2 Then');
fprintf(fid,'%s\n','rhino.OrientObject arc, Array(startp,
endp), Array(arrPt2, arrPt1)');

fprintf(fid,'%s\n','Else');
fprintf(fid,'%s\n','rhino.OrientObject arc, Array(startp,
endp), Array(arrPt1, arrPt2)');
fprintf(fid,'%s\n','End If');
    fprintf(fid,'%s\n','del = Array(curve, p)');
fprintf(fid,'%s\n','rhino.DeleteObjects del');
fprintf(fid,'%s\n','build_arc = arc');
fprintf(fid,'%s\n','End Function');
fclose(fid);
return
```

A.1.5 Editing script for calcium

```
function write_script_calcio(name_file,data_ca,elems_ca,np_ca)
fid = fopen(name_file,'wt');

%%%% IN
fprintf(fid,'%s\n', '*Node');

i=[];
for i=1:np_ca
         fprintf(fid,'%d %s %d %s %d %s
         %d\n',i,',',data_ca(i,1),',', data_ca(i,2),','
         ,data_ca(i,3));
end
fprintf(fid,'%s\n', '*Element, type=S3R');
for i=1:size(elems_ca,1)
         fprintf(fid,'%d %s %d %s %d %s
```

```
        %d\n',i,',',elems_ca(i,1),',', elems_ca(i,2),','
        ,elems_ca(i,3));
end
fclose(fid);
```

Appendix B

MATLAB codes for CoreValve model

B.1 Rotation of elementary unit mesh in a polar series

```
clear all
close all
clc

load('nodes.txt');
load('elem.txt');
load('angles.txt');

nnod = size(nodes,1);
nel = size(elem,1);
na = length(angles);

nrot = na + 1;
ind_err=nodes(:,1);
nodes(:,1)=1:nnod;

nodes_tot = zeros(nnod,4,nrot);
nodes_tot(:,:,1) = nodes;
```

```
for h=2:9
    for k=1:nel
        ind=find(ind_err==elem(k,h));
        elem(k,h)=ind;
    end
end

el_tot = zeros(nel,9,nrot);
el_tot(:,:,1) = elem;

figure,
plot3(nodes(:,2),nodes(:,3),nodes(:,4),'*b')
grid on
hold on

for a=1:na


    for i=1:nnod

        phi = angles(a)*pi/180;
        Rz  = [cos(phi) sin(phi) 0; -sin(phi) cos(phi) 0; 0 0 1];

        nodes_tot(i,1,a+1)   = nnod*a+nodes_tot(i,1,1);
        nodes_tot(i,2:4,a+1) = (Rz * nodes(i,2:4)')';

    end

    for e=1:nel

        el_tot(e,1,a+1) = nel*a+el_tot(e,1,1);
        el_tot(e,2:9,a+1) = nnod*a+elem(e,2:9);

    end

end

name_file = 'Corevalve_no_merge';
```

```
write_corevalve(nnod,nel,na,nodes_tot,el_tot,name_file);
```

B.1.1 Editing INP file for Abaqus

```
function write_corevalve(nnod,nel,na,nodes_tot,el_tot,name_file);

fid = fopen(name_file,'wt');

fprintf(fid,'%s\n', '*Node');

for a=1:na+1
    for i=1:nnod
        fprintf(fid,'%d %s %d %s %d %s %d\n',...
        nodes_tot(i,1,a),', ', nodes_tot(i,2,a), ', ',
        nodes_tot(i,3,a), ', ', nodes_tot(i,4,a));
    end
end

fprintf(fid,'%s\n', '*Element, type=C3D8R');

for a=1:na+1
    for e=1:nel
        fprintf(fid,'%d %s %d %s %d %s %d %s %d %s
        %d %s %d %s %d %s %d\n', el_tot(e,1,a), ', ',...
        el_tot(e,2,a), ', ', el_tot(e,3,a), ', ',...
        el_tot(e,4,a), ', ', el_tot(e,5,a),...
        ', ', el_tot(e,6,a), ', ', el_tot(e,7,a), ', ',
        el_tot(e,8,a), ', ', el_tot(e,9,a));
    end
end


fclose(fid);
```

Bibliography

[1] Brian H. Grimard and Jan M. Larson. Aortic stenosis: Diagnosis and treatment. *American Family Physician*, 2008.

[2] Ross J Jr and Braunwald E. Aortic stenosis. *Circulation*, 1968.

[3] Daria Cosentino Giorgia M. Bosi Silvia Schievano, Claudio Capelli and Andrew M. Taylor. *Finite Element Analysis to Study Percutaneous Heart Valves- From Biomedical Applications to Industrial Developments*, chapter 7, pages 168–192. In Tech, 2012.

[4] Simone Morganti. *Finite Element Analysis of Aortic Valve Surgery*. PhD thesis, 2008-2011.

[5] S. Morganti P. Totaro F. Auricchio, M. Conti. A computational tool to support pre-operative planning of stentless aortic valve implant. *Medical Engineering and Physics*, 2011.

[6] F. Auricchio, M. Conti, S. Morganti, and A. Reali. Simulation of transcatheter aortic valve implantation: a patient-specific finite element approach. *Computer Methods in Biomechanics and Biomedical Engineering*, 2013.

[7] FACC Jean C. Laborde 2 MD Bernfried Zickmann 3 MD Ulrich Gerckens 1 MD Thomas Felderhoff 4 MD Barthel Sauren 3 MD Andreas Bootsveld 1 MD Lutz Buellesfeld 1 MD Eberhard Grube, 1* MD and 4 MD Stein Iversen. First report on a human percutaneous transluminal implantation of a self-expanding valve prosthesis for interventional treatment of aortic valve stenosis. *Catheterization and Cardiovascular Interventions 66:465–469 (2005)*, 2005.

[8] M. J. Thrubrikar. The aortic valve. *CRC Press*, 1990.

[9] Bach, Siao, Girard, Duvernoy, McCallister, and Gualano. Evaluation of patients with severe symptomatic aortic stenosis who do not undergo aortic valve replacement. *Circ Cardiovasc Qual Outcomes*, 2009.

[10] Craig R. Smith M.D. Michael Mack M.D. D. Craig Miller M.D. Jeffrey W. Moses M.D. Lars G. Svensson M.D. Ph.D. E. Murat Tuzcu M.D. John G. Webb M.D. Gregory P. Fontana M.D. Raj R. Makkar M.D. David L. Brown M.D. Peter C. Block M.D. Robert A. Guyton M.D. Augusto D. Pichard M.D. Joseph E. Bavaria M.D. Howard C. Herrmann M.D. Pamela S. Douglas M.D. John L. Petersen M.D. Jodi J. Akin M.S. William N. Anderson Ph.D. Duolao Wang Ph.D. Martin B. Leon, M.D. and for the PARTNER Trial Investigators Stuart Pocock, Ph.D. Transcatheter aortic-valve implantation for aortic stenosis in patients who cannot undergo surgery. *The new england journal of medicine*, 2010.

[11] Andersen, Knudsen, and Hasenkam. Transluminal implantation of artificial heart valves: description of a new expandable aortic valve and initial results with implantation by catheter technique in closed chest pigs. *Eur Heart J.*, 1992.

[12] Alain Cribier, Helene Eltchaninoff, Assaf Bash, Nicolas Borenstein, Christophe Tron, Fabrice Bauer, Genevieve Derumeaux, Frederic Anselme, François Laborde, and Martin B. Leon. Percutaneous transcatheter implantation of an aortic valve prosthesis for calcific aortic stenosis first human case description. *Circulation*, 2002.

[13] Eberhard Grube, Jean C. Laborde, Ulrich Gerckens, Thomas Felderhoff, Barthel Sauren, Lutz Buellesfeld, Ralf Mueller, Maurizio Menichelli, Thomas Schmidt, Bernfried Zickmann, Stein Iversen, and Gregg W. Stone. Percutaneous implantation of the corevalve self-expanding valve prosthesis in high-risk patients with aortic valve disease the siegburg first-in-man study. *Eur Heart J.*, 2006.

[14] John G. Webb, Sanjeevan Pasupati, Karin Humphries, Christopher Thompson, Lukas Altwegg, Robert Moss, Ajay Sinhal, Ronald G. Carere, Brad Munt, Donald Ricci, Jian Ye, Anson Cheung, and Sam V. Lichtenstein. Percutaneous transarterial aortic valve replacement in selected high-risk patients with aortic stenosis. *Circulation, American Heart Association*, 2007.

[15] Nicolas Chronos MD Stanton J. Rowe BA Rob Michiels MBA Bernard E. Lyons PhD Martin B. Leon MD James I. Fann, MD and MD Aaron

V. Kaplan. Evolving strategies for the treatment of valvular heart disease: Preclinical and clinical pathways for percutaneous aortic valve replacement. *Catheterization and Cardiovascular Interventions*, 2008.

[16] Gurvitch R Ajlan AM Labounty TM Min JK. Leipsic J, Hague CJ. Mdct to guide transcatheter aortic valve replacement and mitral valve repair. *Cardiology Clinics*, 2012.

[17] FACC * Gerhard Schuler MD FACC † Lutz Buellesfeld MD * Ulrich Gerckens MD * Axel Linke MD † Peter Wenaweser MD * Barthel Sauren MD * Friedrich-Wilhelm Mohr MD † Thomas Walther MD † Bernfried Zickmann MD * Stein Iversen MD * Thomas Felderhoff MD * Raymond Cartier MD ‡ Raoul Bonan MD FACC‡ Eberhard Grube, MD. Percutaneous aortic valve replacement for severe aortic stenosis in high-risk patients using the second- and current third- generation self-expanding corevalve prosthesis. *Journal of the American College of Cardiology*, 2007.

[18] Mogtader A Mindich B Thys D Darmon PL, Hillel Z. Cardiac output by transesophageal echocardiography using continuous-wave doppler across the aortic valve. *Anesthesiology*, 1994.

[19] J. Yapc M.J. Mullenc G. Burriescia S. Tzamtzisa, J. Viqueratb. Numerical analysis of the radial force produced by the medtronic-corevalve and edwards-sapien after transcatheter aortic valve implantation (tavi). *Medical Engineering and Physics*, 2013.

[20] E. Cerri J. Nordmeyer T. Odenwald P. Bonhoeffer F. Migliavacca A. M. Taylor S. Schievano C. Capelli, G. M. Bosi. Patient-specific simulations of transcatheter aortic valve stent implantation. *International Federation for Medical and Biological Engineering*, 2012.

[21] Ulrich Gerckens Thomas Felderhoff Barthel Sauren Lutz Buellesfeld Ralf Mueller Maurizio Menichelli Thomas Schmidt Bernfried Zickmann SteinIversen Eberhard Grube, Jean C. Laborde and Gregg W. Stone. High-risk patients with aortic valve disease: The siegburg first-in-man study percutaneous implantation of the corevalve self-expanding valve prosthesis in. *Circulation*, 2006.

[22] C.A. Basciano S. Seelecke M.A. Farber C. Kleinstreuer, Z. Lic. Computational mechanics of nitinol stent grafts. *Journal of Biomechanics*, 2008.