

INAUGURAL – DISSERTATION
zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der
Ruprecht-Karls-Universität
Heidelberg

vorgelegt von
Dipl.-Math. techn. Nicolai Schoch
aus Lauterbach

Tag der mündlichen Prüfung:

Towards Cognition-Guided Patient-Specific Numerical Simulation for Cardiac Surgery Assistance

Betreuer: Prof. Dr. Vincent Heuveline

To my parents

Abstract (English)

Motivation. Patient-specific, knowledge-based, holistic surgical treatment planning is of utmost importance when dealing with complex surgery. Surgeons need to account for all available medical patient data, keep track of technical developments, and stay on top of current surgical expert knowledge to define a suitable surgical treatment strategy. There is a large potential for computer assistance, also, and in particular, regarding surgery simulation which gives surgeons the opportunity not only to plan but to simulate, too, some steps of an intervention and to forecast relevant surgical situations.

Purpose. In this work, we particularly look at *mitral valve reconstruction* (MVR) surgery, which is to re-establish the functionality of an incompetent mitral valve (MV) through implantation of an artificial ring that reshapes the valvular morphology. We aim at supporting MVR by providing surgeons with biomechanical FEM-based MVR surgery simulations that enable them to assess the simulated behavior of the MV *after* an MVR. However, according to the above requirements, such surgery simulation is really beneficial to surgeons only if it is patient-specific, surgical expert knowledge-based, comprehensive in terms of the underlying model and the patient's data, and if its setup and execution is fully automated and integrated into the surgical treatment workflow.

Methods. This PhD work conducts research on *simulation-enhanced, cognition-guided, patient-specific cardiac surgery assistance*. First, we derive a biomechanical MV/MVR model and develop an FEM-based MVR surgery simulation using the FEM software toolkit HiFlow³. Following, we outline the functionality and features of the Medical Simulation Markup Language (MSML) and how it simplifies the biomechanical modeling workflow. It is then detailed, how, by means of the MSML and a set of dedicated MVR simulation preprocessing operators, patient-individual medical data can comprehensively be analyzed and processed in order for the fully automated setup of MVR simulation scenarios. Finally, the presented work is integrated into the cognitive system architecture of the joint research project *Cognition-Guided Surgery*. We particularly look at its semantic knowledge and data infrastructure as well as at the setup of its cognitive software components, which eventually facilitate cognition-guidance and patient-specificity for the overall simulation-enhanced MVR assistance pipeline.

Results and Discussion. We have proposed and implemented, for the first time, a prototypic system for simulation-enhanced, cognition-guided, patient-specific cardiac surgery assistance. The overall system was evaluated in terms of functionality and performance. Through its cognitive, data-driven pipeline setup, medical patient data and surgical information is analyzed and processed comprehensively, efficiently and fully automatically, and the hence set-up simulation scenarios yield reliable, patient-specific MVR surgery simulation results. This indicates the system's usability and applicability. The proposed work thus presents an important step towards a simulation-enhanced, cognition-guided, patient-specific cardiac surgery assistance, and can – once operative – be expected to significantly enhance MVR surgery. Concluding, we discuss possible further research contents and promising applications to build upon the presented work.

Abstract (German)

Motivation. Eine patientenspezifische, wissensbasierte, gesamtheitliche Chirurgie-Behandlungsplanung ist bei komplexen Operationen von höchster Bedeutung. Chirurgen müssen dafür alle verfügbaren medizinischen Patientendaten berücksichtigen, sowie aktuelle technische Entwicklungen kennen, und einen Überblick über relevantes chirurgisches Expertenwissen besitzen. Dies ergibt ein großes Potential für IT-basierte Unterstützung, auch – und insbesondere – im Hinblick auf Chirurgiesimulationen, welche es Chirurgen ermöglichen, gewisse Schritte einer Operation nicht nur zu planen, sondern diese auch zu simulieren, und so relevante chirurgische Situationen hervorzusagen.

Zweck. In dieser Arbeit wird die chirurgische Mitralklappenrekonstruktion (MKR) betrachtet, welche die Funktionalität einer inkompetenten Mitralklappe (MK) durch Implantieren eines künstlichen Ringes, der die Klappenmorphologie verändert, wiederherstellen soll. Ziel ist es, Chirurgen bei MKR durch biomechanische FEM-Simulationen zu unterstützen, und sie so dazu zu befähigen, das simulierte MK-Verhalten *nach* einer potentiellen MKR *präoperativ* zu untersuchen. Hierzu müssen derartige Simulationen jedoch patientenspezifisch und im Bezug auf das biomechanische Modell sowie auf die Patientendaten ganzheitlich aufgesetzt sein. Außerdem sollte ihre Ausführung vollständig automatisiert und in den chirurgischen Behandlungsablauf eingebettet geschehen.

Methoden. Diese Doktorarbeit erforscht simulationsbasierte und kognitionsgeleitete, patientenspezifische Herzchirurgieassistenz. Zunächst wird ein biomechanisches MK-Modell hergeleitet, sowie eine MKR-Simulation unter Verwendung der FEM-Software HiFlow³ entwickelt. Weiter wird die Medical Simulation Markup Language (MSML) beschrieben und gezeigt, wie diese den biomechanischen Modellierungsworkflow vereinfacht. Basierend auf der MSML und mittels einer Reihe von speziellen Preprocessing-Operatoren werden Patientendaten analysiert und weiterverarbeitet, so dass daraus vollautomatisch MKR-Simulationsszenarien generiert werden können. Schließlich wird die Arbeit in die kognitive Systemarchitektur des SFB *Cognition-Guided Surgery* integriert. Dabei liegt der Fokus auf der semantischen Wissens- und Daten-Infrastruktur und auf dem Setup von kognitiven Software-Komponenten, welche letztlich Cognition-Guidance und Patientenspezifität für die simulationsbasierte MKR-Assistenz ermöglichen.

Ergebnisse und Diskussion. Erstmalig wurde ein prototypisches System zur kognitionsgeleiteten und simulationsbasierten, patientenspezifischen Herzchirurgieassistenz vorgestellt. Es wurde im Hinblick auf Funktionalität und Performance evaluiert. Dank seines kognitiven, datengetriebenen Pipeline-Setups werden medizinische Patientendaten umfassend, effizient und vollautomatisch analysiert und weiterverarbeitet. Die daraus generierten Simulationsszenarien ermöglichen das Berechnen von patientenspezifischen MKR-Simulationsergebnissen. Dies deutet auf die Eignung des Systems hin. Die vorgestellte Arbeit stellt somit einen Schritt in Richtung kognitionsgeleiteter und simulationsbasierter, patientenspezifischer Herzchirurgieassistenz dar, und es kann davon ausgegangen werden, dass sie die MKR-Chirurgie wesentlich aufwerten wird. Abschließend werden mögliche, aufbauende Anwendungen und zukünftige Forschungsfragen diskutiert.

Acknowledgements

Above all, I thank my parents and sister for their love and for their steady support throughout my whole life; for just being there and for being such a unique, lovely family without whom I could not live.

I thank my girlfriend Eda and my friends, Jonas, Timo, and so many more, for their love and friendship, for being there when I needed them, for their time and for the time together, and for supporting me, for re-motivating me, for distracting me when work would make too big demands, and for forgiving me after being uncomfortable when work wouldn't work every once in a while.

I want to express my gratitude to Prof. Dr. Vincent Heuveline, who offered me the possibility to join his research group and to immerse myself in this fascinating research on numerical simulation and cognition-guidance for surgery assistance. I thank him for his expert guidance and encouragement as well as many, many insightful discussions for the time of my PhD and for the future. Also, I am very thankful to Prof. Dr. Götz Alefeld for his time and efforts, for his valuable feedback and for fruitful, inspiring and pleasant discussions on my thesis work.

Further, I want to thank all colleagues and members of the EMCL, HITS and URZ groups, for an amazing, great time and a really cool working atmosphere, as well as for valuable discussions, inspiration, feedback, coffee and cookie breaks, and for the unique EMCL spirit.

I thank the DFG for funding the innovative, trend-setting and socially extremely relevant research conducted in the framework of the collaborative research center *Cognition-Guided Surgery* (SFB TRR 125), as well as all SFB members and associates for their friendly, helpful and cooperative attitude and for fascinating, inspiring research results.

In this context, my special thanks go to Stefan Suwelack and Markus Stoll (in the MSML project), Sandy Engelhardt, Prof. Dr. Ivo Wolf, and Prof. Dr. Raffaele de Simone (with respect to cardiac surgery and image segmentation), Patrick Philipp and Prof. Dr. York Sure-Vetter (in what concerns Cognitive Applications and semantics), Steffi Speidel (in assistance during my first publication), and the student research workers Fabian Kießler and Alexander Weigl (in the context of MSML and MVR simulation preprocessing).

Contents

1	Introduction and Motivation	1
1.1	Cognition-Guided Surgery	3
1.2	Cognitive Surgery Simulation and the Motivation of this Work	4
2	Clinical Application Scenario: Cardiac Surgery	7
2.1	Anatomy and Functionality of the Human Heart	7
2.2	MV Pathophysiology and Surgical Treatment	11
3	Cardiac Surgery Assistance: State-of-the-Art and Current Open Challenges	15
3.1	State-of-the-Art of Simulation-based Cardiac Surgery Assistance Workflows	15
3.1.1	Image Acquisition	16
3.1.2	Image Processing and Image Segmentation	17
3.1.3	Segmentation Postprocessing and 3D Modeling	18
3.1.4	General Information Processing, Knowledge Representation and Data Science in Surgery Assistance Environments	20
3.1.5	Biomechanical Modeling and Surgery Simulation	24
3.2	Open Challenges in the Context of Simulation-based Cardiac Surgery Assistance	36
3.3	Concept for Cognition-Guided, Simulation-enhanced Cardiac Surgery Support	38
3.3.1	Requirements for Cognition-guided, Simulation-based Surgery Assistance	40
3.3.2	Context, Goals and Structure of this Work	41
4	Modeling and Simulating the Behavior of the Mitral Valve	43
4.1	Basics of Elasticity Theory	43
4.1.1	Kinematics and the Concepts of Deformation, Strain and Stress	43
4.1.2	Equilibrium Equations and the Boundary Value Problem of Elasticity	48
4.1.3	Constitutive (Material) Laws for Biological Soft Tissue	51
4.1.4	Contact Boundary Value Problem and Variational Form	70
4.2	Modeling the Elastic Behavior of the Mitral Valve	75
4.2.1	Derivation and Specification of the General Elasticity Formulation for Modeling and Simulation of the MV and MVR	75
4.2.2	MV Modeling and Simulation: Current Limitations	85
4.3	Numerical Solution of the Contact Elasticity Problem for the Mitral Valve	87
4.3.1	Discretization in Space – The Finite Element Method (FEM)	87
4.3.2	Discretization in Time – RK4 and Newmark Time Integration	92
5	MVR simulation using the FEM	101
5.1	General Setup of an FEM-based Simulation	101
5.1.1	FEM Software HiFlow ³	101

5.1.2	General FEM Code Structure	102
5.2	Setup of our FEM-based MVR Elasticity Simulation	105
5.2.1	Analysis of the Code Structure, Code Control Mechanisms & Interfaces	105
5.2.2	Code Optimization and High-Performance Computing (HPC)	113
5.2.3	Outlook and Transition to the next two Chapters	118
6	Biomechanical Model and Simulation Management by means of the MSML	121
6.1	The Medical Simulation Markup Language (MSML)	121
6.2	Comprehensive Simulation Preprocessing for MVR	129
6.2.1	Setting up a Chain of Operators for MVR Simulation Preprocessing	131
6.2.2	Integration of the MVR-specific Operators into the MSML	142
6.3	Simulation Postprocessing for MVR Surgery Assistance	147
6.4	Discussion, Future Intended Work and Outlook	151
7	Cognition-Guidance for Simulation-enhanced Cardiac Surgery Assistance	155
7.1	Basics of Clinical Information Processing and Surgical Data Science	155
7.2	Semantic System Architecture for Cognition-Guided Surgery Assistance	162
7.2.1	Semantic Data Infrastructure and Clinical Knowledge Representation	162
7.2.2	Semantic Software Architecture and Cognitive Software Components	169
7.3	Sample Application: Cognitive Tools Pipeline for MVR Surgery Assistance	176
7.4	Discussion and Outlook: Towards Semantic Simulation for Surgery Assistance	184
8	Evaluation of the Overall Prototypic Simulation-enhanced Surgery Assistance Setup	191
8.1	Functional Evaluation of the Overall Surgery Assistance Setup	191
8.2	Numerical Analysis and Performance Evaluation of the MVR Simulation	203
8.3	Outlook with respect to further Evaluation of Model and Simulation	208
9	Summary, Discussion and Outlook	209
9.1	Summary and List of own Contributions	209
9.2	Discussion and Conclusion	212
9.3	Outlook and Future Work	215
9.3.1	Possible Future Applications to Build Upon the Presented Work	215
9.3.2	Possible Ongoing Research and Development	216
A	Appendix – List of Publications	221
	Bibliography	223

1 Introduction and Motivation

Looking at surgical interventions today, during complex surgery, for a successful operation the surgeon is required to keep in mind and consider an enormous number of different factors, such as the patients' current condition, their health history records, previous diseases, anatomically peculiar features, and many more medical and clinical data. Accounting for these masses of potentially important impact factors, and keeping track of the technical development in diagnostic and therapeutic procedures, the surgeons then need to apply their surgical expert knowledge and experience in order to define a suitable surgical treatment strategy and to properly conduct surgery; see Figure 1.1.

There are loads of assistance tools and technical devices in the operation room (OR), from surgical instruments and medical devices, via anesthesia and navigation, to robotics support and somatic functions surveillance; see Figure 1.2. However, the only intelligent, thinking subject in the OR commonly is the surgeon and the surgical staff, who make use of their individual perception, cognition, knowledge, and experience-based interpretation and judgement capability in order to suitably treat patients and their diseases according to their medical data.

Considering the progressing digitalization in various areas of daily life and society, and given the enormous speed and extend of progress in the field of IT developments, one may easily figure the great potential of these advances for surgery and surgery assistance, too, so the above described situation can be improved or enhanced.

As such, the rise of big clinical data bases, and innovative, intelligent information processing algorithms, as well as sophisticated data analysis and reasoning methods along with semantic web technologies have enormously driven the development of computer-assisted surgery over the last few years, and have let, for instance, clinical decision support systems and surgical simulation systems find their way into the OR.

Ongoing, it is obvious, and there is broad general consensus, that this potential must be further exploited. A more comprehensive, holistic, patient-specific surgical treatment planning, as well as an interdisciplinary cooperation across different clinical departments in order for a more complete and each other complementing surgical reasoning procedure seem to be within reach. Additional, possibly *new-in-its-nature*, valuable and important knowledge and information for surgery may thus be derived – preferably patient-individually and fully automated.

Many different research fields are therefore made to cooperate and to thus mutually benefit from each other: medicine and surgery, imaging and image processing, radiology, radiotherapy, medical physics, cognitive sciences, data science, artificial intelligence, semantics, computer sciences for software and hardware development, mathematics, numerical modeling and simulation, robotics, and many more, ... all aiming at an improved *surgery in the future*.

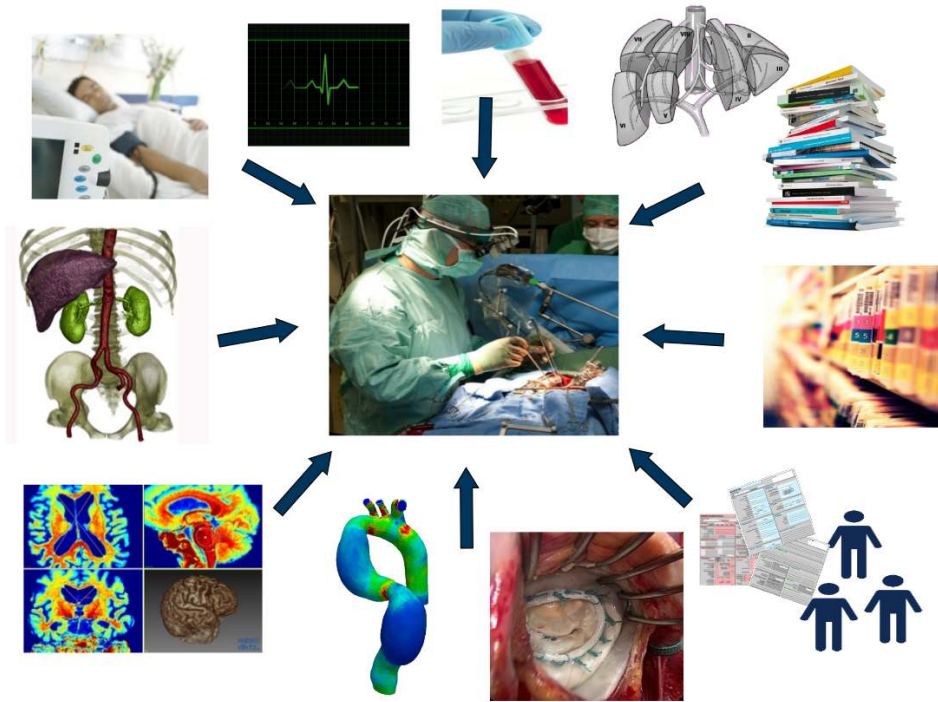


Figure 1.1: During complex surgery, an enormous amount of surgically relevant impact factors and information have to be considered and taken into account by the surgeon in order to derive suitable surgical treatment strategies. [Images partly reprinted and collaged from Wikimedia Commons and from EMCL inhouse data bases.]



Figure 1.2: There is lots of high-end technology and tools in the OR which facilitate access to patient data and to other supporting information. Through these, surgery is getting better and more accurate, however, it is also getting more and more complex.

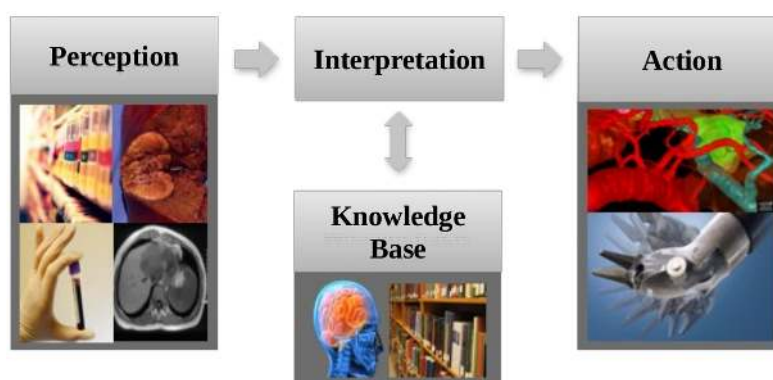


Figure 1.3: Draft of the cognition-guided surgery approach of the transregional collaborative research center *Cognition-Guided Surgery* (SFB/TRR 125). [Illustration adapted and reprinted from <http://www.cognitionguidedsurgery.de/1/home/>.]

1.1 Cognition-Guided Surgery

An improvement of the above described currently common situation in the OR and the consequential need for intelligent, comprehensive, cognitive surgery support is what the interdisciplinary, multi-institutional, transregional collaborative research center *Cognition-Guided Surgery* (SFB TRR 125)¹ is geared towards. The SFB TRR 125 is a joint research effort between the German Cancer Research Center (DKFZ Heidelberg), the University of Heidelberg, the Heidelberg University Hospital (Uniklinikum Heidelberg) and the Karlsruhe Institute of Technology (KIT Karlsruhe). It is funded by the DFG².

An interdisciplinary consortium of surgeons from different clinical disciplines, as well as radiologists, engineers, mathematicians and computer scientists is working together on the vision of a technical-cognitive assistance system that supports surgeons in finding the best possible surgical therapy and treatment for individual patients. In a joint effort, this consortium aims at exploring new methods and possible solutions for knowledge- and model-based, cognition-guided surgery.

The overall SFB project vision of *surgery in the future* is a surgery that is lead by machine cognition, i.e., a surgery where an intelligent surgery support system recognizes clinical situations (*perception*), interprets them (*interpretation*) on the basis of a knowledge data base and through intelligent algorithms (*knowledge base*), and suggests adequate surgical actions to the operating surgeon (*action*). See Figure 1.3 for a draft of the cognition scheme.

As such, the technical-cognitive system supports the surgeon in a way similar to how an intelligent human assistant would do. For instance, the system is to be able to:

- collect and combine preoperative, perioperative and post-operative patient information,
- acquire and incorporate both factual knowledge (e.g., from literature, guidelines, etc.) and practical knowledge (e.g., from experience),

¹SFB TRR 125 *Cognition-Guided Surgery*, see <http://www.cognitionguidedsurgery.de/1/home/>

²DFG: Deutsche Forschungsgemeinschaft, see www.dfg.de

- continuously follow the operational procedure and gather relevant information,
- use all information and knowledge to assess and interpret the current in-OR-situation,
- possibly perform further reasoning and conduct further analysis,
- recommend, when required, an appropriate subsequent action, and
- eventually – and this is to be seen as future work beyond the project focus –, return the results to the knowledge base as part of a learning progress, and store the result information in order to make it available for future use.

Surgery is therefore expected to be facilitated and conducted in a more efficient and qualitative way. Not only can be expected to achieve a safer and better-substantiated surgical treatment as opposed to today's largely instinctive and personal experience-based surgery. The surgical benefits are even more wide-reaching: technology has the potential to support surgeons so they can derive and perform the possibly best suitable operation, which is least invasive and least critical for the patient. This way, lower re-operation rates and lower mortality rates can be achieved, too, and surgical time and cost is saved.

1.2 Cognitive Surgery Simulation and the Motivation of this Work

One idea – besides many others – which perfectly fits together with the vision of *Cognition-Guided Surgery*, and which actually constitutes an essential component of a *cognition-guided surgery assistance system*, is to enhance traditional surgery by means of *surgical simulations*.

Research and development in the context of *intelligent, cognition-guided, patient-individual surgery simulation* for surgery assistance embodies the content of this PhD work, which has been conducted as part of the SFB TRR 125 sub-project I03 on *Functional Modeling and Numerical Simulation* at the *Engineering Mathematics and Computing Lab* (EMCL) at the IWR at Heidelberg University, Germany.

Extending traditional surgery that usually relies on static preoperative image data and on otherwise preoperatively recorded clinical patient information, surgery simulation can provide surgeons with 3D models of the organs that are operated on, and with mathematical measurements and simulations which can be used for diagnostic and risk analysis purposes or for an overall better understanding of the complex biomedical system of the human body. Moreover, surgery simulation can yield additional predictive information, e.g., through simulating surgical interventions *before* these actually take place. Surgeons can hence virtually assess a respective organ's behavior subject to external forces or momentums, before and after a potential surgical manipulation, in 3D, and – as opposed to traditional intraoperative assessment – without time pressure and in an uncritical virtual environment, where trial-and-error with respect to a potentially suitable surgical strategy is not fatal but only virtual, and where optimization in terms of re-operation is possible, etc.

Real benefit for a surgeon through surgery simulation, however, can only be achieved if the simulation scenarios are set-up in a maximally comprehensive and patient-specific way, and if they consider and incorporate all currently available surgical expert knowledge with respect to a potentially best surgical procedure. Yet, in order to have a computer system consider this

knowledge and apply it on an individual patient's data, this requires for a competent semantic knowledge representation and management, as well as for an intelligent, cognitive information processing chain which the aforementioned surgery simulation is a part of. Moreover, to foster the usability of such simulation-based surgery assistance, the respective simulation scenarios need to be set-up through a fully automated workflow, which does not demand for any manual interaction.

Yet, to date, to the best of the author's knowledge, such cognition-guided, simulation-enhanced, patient-individual surgery assistance system has never before been presented or implemented.

In this work, we specifically look at *cardiac surgery* and at *cognition-guided, simulation-enhanced cardiac surgery assistance*. We aim at designing a concept and setting up a first operational prototype for a cognition-guided, simulation-enhanced, patient-individual cardiac surgery assistance system.

Therefore, firstly, the exemplary clinical use case in cardiac surgery will be presented (Chapter 2), and the state-of-the-art in research and development in the respective surgery assistance field will appropriately be outlined (Chapter 3). Thereafter, we will discuss the major research questions and To-Do's in order for the development of such a cognition-guided, simulation-based surgery assistance system, which – just like a human intelligent surgical assistant – supports cardiac surgery by providing the operating surgeon fully-automatically with patient-specific, situation-adapted, expert-knowledge-based, comprehensive surgery simulation scenario results, which are based on a preceding holistic analysis of the available patient information and formalized surgical knowledge (Section 3.3.1). Concluding, we will then present a *detailed structure* of how *this PhD thesis* and the associated work is set-up (Section 3.3.2).

2 Clinical Application Scenario: Cardiac Surgery

In this chapter, in Section 2.1, we first give a short introduction to the anatomy and functionality of the human heart and of the mitral valve in particular. Second, we describe the pathophysiological circumstances which require surgical treatment, and especially focus on the procedure of a mitral valve reconstruction (MVR), see Section 2.2. The anatomical and medical knowledge presented in this chapter is primarily based on [Iai09] and [CAF10].

2.1 Anatomy and Functionality of the Human Heart

The heart is a muscular organ, which pumps blood through the blood vessels of the body's circulatory system, and thus not only provides the body with oxygen and nutrients, but also assists it in the removal of metabolic wastes. The human heart contains four chambers: the upper left and right atria, and lower left and right ventricles, see Figure 2.1.

In a healthy heart, by means of repeated rhythmic contraction, blood is pumped through these chambers. Due to the heart valves, which in healthy condition only allow for oneway-directed flow, the blood flows oneway through the heart, and a backflow of blood (so-called regurgitation) is prevented, see Figure 2.1.

The heart pumps blood through the body. Blood that is low in oxygen from the systemic circulation through the body enters the heart from the superior and inferior vena cavae, and passes due to the blood pressure gradient from the right atrium through the *tricuspid valve* into the right ventricle. From here, by means of contraction of the heart muscle, it is pumped through the *pulmonary valve* into the pulmonary circulation system, i.e., through the lungs, where an exchange of gases occurs through the passive process of diffusion: it gives off carbon dioxide (CO_2) and receives oxygen (O_2). Oxygenated blood then returns to the heart, which it enters through the left atrium. Again due to a pressure gradient, it passes through the *mitral valve* to the left ventricle, and is pumped by means of the contracting heart muscle through the *aortic valve* out through the aorta to the systemic circulation. Here, the oxygen is used and supplies organs and tissue. It thus gets metabolized to carbon dioxide, and the whole circle starts again.

Two phases in this *heart cycle* are commonly distinguished: The *systole* is the phase in which the heart and its ventricles are contracting and thus pushing the blood through the open pulmonary and aortic valves out of the ventricles (while mitral and tricuspid valve are closed). The *diastole* is the phase in which the ventricles are dilating and thus allowing for the blood to stream through the open mitral and tricuspid valves into the ventricles (while pulmonary and aortic valve are closed). Obviously, the proper opening and closing functionality of all four heart valves is essential for the directed blood circulation, and thus for the human health.

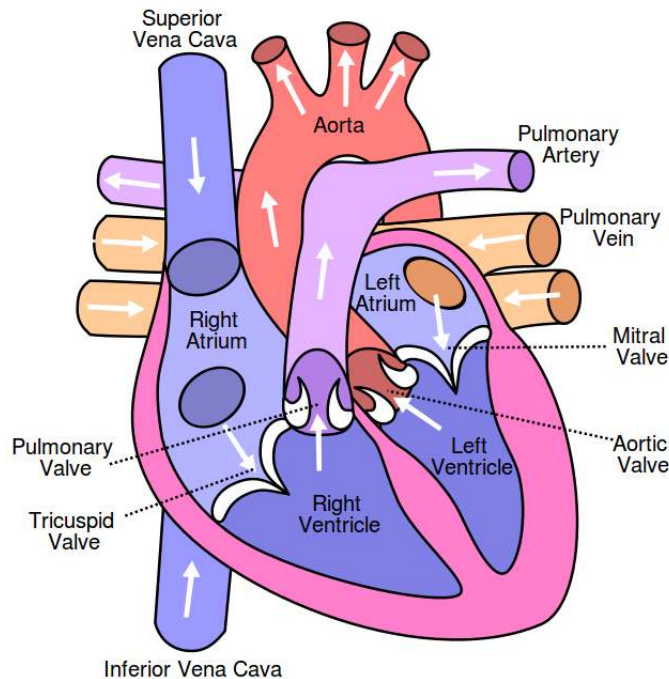


Figure 2.1: Draft of the human heart and its four chambers and valves. [Illustration reprinted from Wikimedia Commons.]

Of the four heart valves, the *mitral valve* (MV) is the most complex one. It regulates the forward flow of blood between left atrium and left ventricle. The complete valvular apparatus – see Figure 2.2 – consists of the following four principal substructures:

- *The mitral annulus*, a saddle-shaped ring of thickened, rigid connective tissue, which provides essential stabilization to the valve and supports the two leaflets;
- *The two leaflets*, the anterior and posterior leaflet, which are pushed down to allow the flow of blood from the left atrium into the left ventricle during diastole, and which are moved up during systole, when by the contraction of the ventricle the blood is pushed into scallops attached to the leaflets, causing them to close;
- *The chordae tendinae*, a complex web of branching fibrous chords, which anchor the two leaflets to two muscular promontories of the left ventricular wall, the anterior and posterior papillary muscles;
- *The two papillary muscles*, which are extensions of and supported by the muscle of the ventricular wall.

The MV closing procedure starts with the contraction of the ventricle in systole, which causes the blood to run into the leaflets' scallops and thus makes the leaflets move up. Through the synergetic action of the above four valve substructures the valve then closes and a backflow of blood from the left ventricle into the left atrium is prevented: The papillary muscles contract

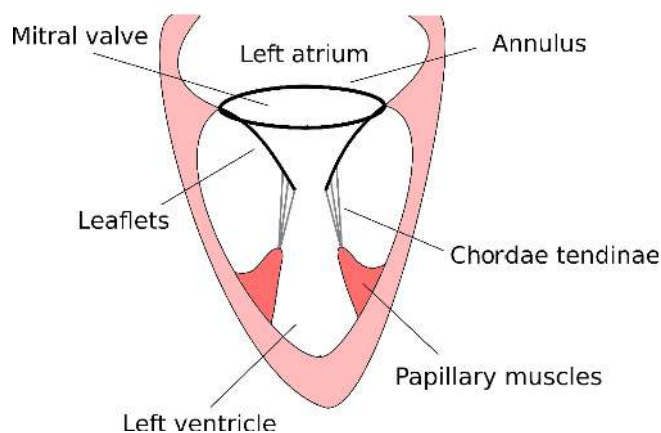


Figure 2.2: Draft of the left ventricle of the human heart and the mitral valve apparatus.

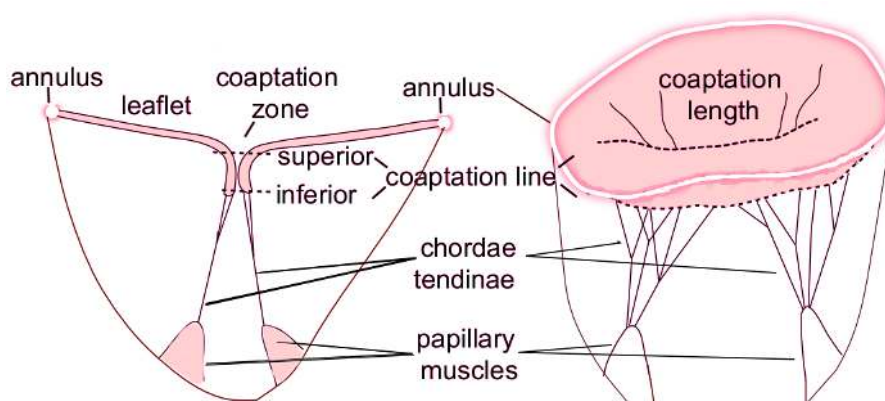


Figure 2.3: Draft of the mitral valve apparatus and the formation of the coaptation zone.

and tighten the chordae tendinae, which in turn regulate the leaflets dynamics and prevent them from over-opening and prolapsing into the left atrium, see Figure 2.3. Besides that, the synergetic action ensures proper *leaflets coaptation*, which describes the situation when the two leaflets get in contact with each other and thereby close the gap (or throughflow tract) in between them. Once closed, a *coaptation zone* is formed between the two leaflets, see Figure 2.3. The bigger this contact zone, the better the valve can potentially beware from leakage. The so-called superior and inferior *coaptation lines* then describe the visible upper and lower edges of the coaptation zone; the upper one ends in the two so-called commissure points.

The respective schematic courses of physiologically important quantities – namely, the ventricular volume, as well as the atrial and ventricular pressure – over time during a cardiac cycle can be seen along with the associated electro- and phono-cardiogram in the so-called *Wiggers Diagram*; see Figure 2.4.

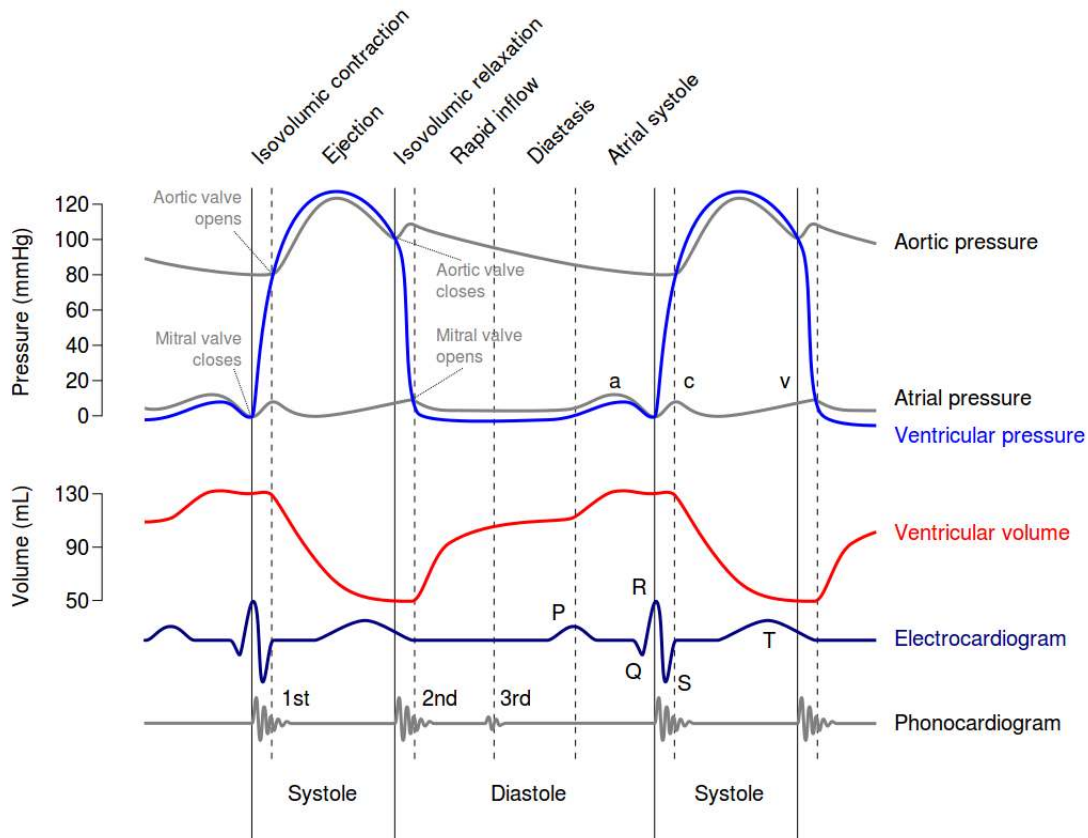


Figure 2.4: A Wiggers Diagram, showing the cardiac cycle events occurring in the left ventricle. In the *atrial pressure plot*, wave **a** corresponds to atrial contraction, wave **c** corresponds to an increase in pressure from the MV bulging into the atrium after closure, and wave **v** corresponds to the passive atrial filling. In the *electrocardiogram*, wave **P** corresponds to atrial depolarization, waves **QRS** correspond to ventricular depolarization, and wave **T** corresponds to ventricular repolarization. In the *phonocardiogram*, the sound labeled **1st** is the reverberation of blood from the sudden closure of the MV, and the sound labeled **2nd** is the reverberation of blood from the sudden closure of the AV. [Illustration reprinted from Wikimedia Commons.]

2.2 MV Pathophysiology and Surgical Treatment

The mitral valve is not only the most complex valve, but also the one most associated with disease [Tur04]. Based on a recent study performed in the United States on a population of almost 12,000 patients, valvular heart disease (VHD) prevalence and incidence is estimated to amount to approximately 2.5%, with a progressive increase with patients' age, up to 13.2% after 75 years of age [NGT⁺06]. Among the four heart valves, the aortic valve (AV) and the mitral valve (MV) are the most frequently affected by pathologies requiring surgical intervention; mostly through replacement for the AV and through repair for the MV, cf. [BA09] and [GSG⁺09].

A number of processes may affect the proper functionality of the MV and lead to regurgitant flow of blood from the left ventricle into the left atrium, see Figure 2.5. This reflux of blood limits the pump function of the heart and among others leads to blood congestion in the lungs. The corresponding pathological condition is known as *mitral regurgitation* (MR). MR is subcategorized into disease processes intrinsic to the valve, which affect its functionality by altering the leaflets, chordae, and mitral annulus (so-called primary MR), and those processes extrinsic to the valve, which cause a deformation of the left ventricle itself, and which lead to an altered mitral geometry (so-called secondary or functional MR).

Primary MR may emerge after severe forms of valvular degeneration, known as Barlow disease, but can also be a result of various congenital connective tissue disorders, such as Marfan syndrome. Secondary MR may be caused by coronary artery disease or by some myocardial infarction, which often leads to restricted leaflet motion, poor leaflet coaptation and regurgitant blood flow. Figure 2.6 shows an example of MR due to so-called leaflet prolapse, which may be due to elongation or rupture of the chordae or the papillary muscles. The surgical treatment of primary MR therefore tends to be more curative than treatment of secondary MR, which usually is more difficult, too, and often requires for future reoperation(s) [NOB⁺14].

When a surgical intervention is inevitable in very severe cases of MR, the incompetent or dysfunctional valve can either be replaced or reconstructed. Several studies (see, e.g., [JHPH07] or [MHG⁺03]) show that mitral valve reconstruction or repair is preferable over replacement, as it produces the better long-term results with less often required future reoperations.

In his fundamental work on mitral valve reconstructive surgery [CAF10], Alain Carpentier defines three operation principles:

- Preserve or restore full leaflet motion;
- Create or allow for a large coaptation surface;
- Remodel and stabilize the entire mitral annulus.

Consequently, mitral valve reconstruction (MVR) surgery can – depending on the specific category and type of the MR – involve the application of methods for reattaching ruptured chordae or for removing excessive leaflet tissue. Most often, however, according to a study with almost 33,000 mitral valve repair operations, in over 97% of all cases an *annuloplasty* was performed [GSG⁺09]. The goal of an annuloplasty is to re-shape the mitral annulus by means of the implantation of an artificial prosthetic *annuloplasty ring* onto the natural mitral annulus, see Figure 2.7.

The artificial annuloplasty rings are implanted in order to enlarge, shrink or stabilize the natural mitral annulus. Carpentier first introduced the concept of annuloplasty ring implantation

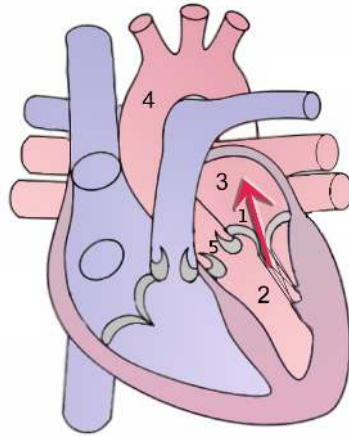


Figure 2.5: Draft of the occurrence of mitral regurgitation. During systole, in a healthy heart the blood is pumped from the left ventricle (2) through the open aortic valve (5) into the aorta (4) and thus into the systemic circulation through the body, while the mitral valve (1) is closed. Mitral regurgitation describes the pathological condition when the mitral valve (1) does not close properly and thus cannot prevent the backflow of blood from the left ventricle (2) into the left atrium (3). [Illustration adapted and reprinted from Wikimedia Commons.]

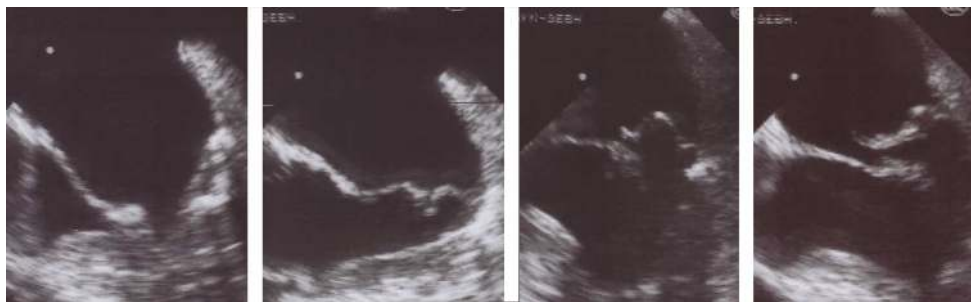


Figure 2.6: Transesophageal echocardiogram (TEE ultrasound image) of a patient suffering from mitral insufficiency, showing the occurrence of an MV leaflet prolapse. The echocardiogram is commonly used to confirm the diagnosis of mitral insufficiency. Color Doppler 4D Ultrasound imaging can additionally reveal a jet of blood flowing from the left ventricle back into the left atrium during ventricular systole (mitral regurgitation), compare Figure 2.5. [Image reprinted from Wikimedia Commons.]

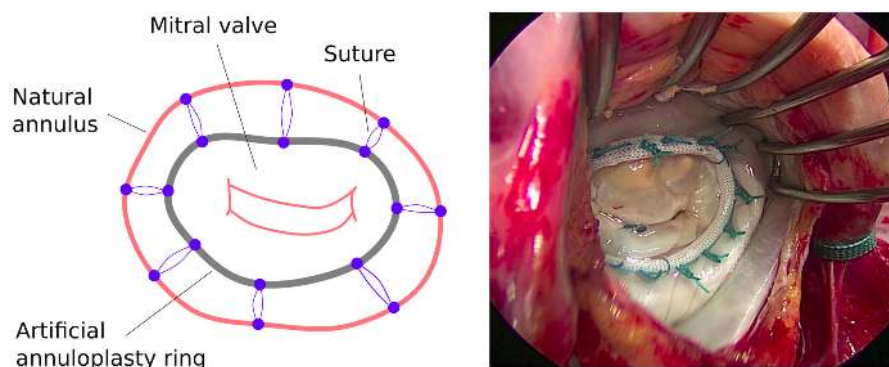


Figure 2.7: (Left) Draft of the surgical implantation of an annuloplasty ring onto the mitral valve. (Right) Endoscopic image taken during an annuloplasty, showing the implantation of a prosthetic annuloplasty ring onto the natural mitral annulus.



Figure 2.8: A set of commercially available annuloplasty rings. [Illustration reprinted with friendly permission from B. Graser [GWAM⁺14].]

in 1968 [CAF10], and since then, many changes to shape and material of these prostheses have been made. Today, a variety of annuloplasty rings is commercially available, see Figure 2.8 for a set of widely employed samples.

There are many different *types* of rings: open and closed ones, flat and saddle-shaped ones, rigid or flexible ones, etc., each of them designed to address various specific pathologies. Besides the existence of different ring types, the respective rings additionally vary in *size*. A precise, scientific measurement of the mitral annulus can provide essential information for the ring size selection, however, the commonly applied measurement techniques (e.g., using a so-called sizer for visual assessment) are prone to errors [BMD13]. Moreover, surgical assessment, which is commonly conducted intraoperatively, at an asystolic, flaccid heart on cardiopulmonary bypass, does not completely and accurately represent any time of the cardiac cycle, i.e., under physiologic loading conditions [EdSAM⁺16]. The actual anatomy can therefore only be properly captured by means of preoperative image acquisition, which in turn is often inaccurate and subject to noise and imaging artefacts, too. Consequently, to date, there is no common base

for correct measurement techniques yet, such that the different ring sizing strategies cannot be compared either.

Hence, in order to accomplish the difficult task of a reasonable ring selection with respect to both type and size, the operating surgeon is required to have profound expert knowledge and experience. Most experts agree that improper sizing and inadequate ring selection during mitral valve reconstruction results in a suboptimal clinical outcome. In the worst case, this can even lead to an incompetent valve and to a recurrence of MR, which eventually implies the necessity of a reoperation. For instance, studies report shockingly high recurrence rates of severe MR after MVR surgery for functional ischemic MR (25% after 6 months) [MGB⁺04] and degenerative valve disease (29% after 7 years) [FHB03]. Besides valve leakage, high tissue tension due to an inappropriate ring prosthesis can cause rupture of the MV tissue and a dehiscence of the ring with life threatening consequences [CAF10, BMD13].

Nevertheless, the surgeons' ideas and judgements about the best ring type selection and ring sizing strategies for the individual patient or pathology do not always coincide. There even are partly non-congruent and conflicting expert opinions (see, e.g., [FMB08, CAF10, AAR⁺06]), and best surgical practices often are chosen and applied without clear scientific evidence, but rather according to the personal experience of the operating surgeon. Moreover, new surgical techniques are continuously being developed, such that surgical success is often hampered as the operating surgeons cannot keep track of them all, not to mention that they cannot consider all potentially available data simply due to time constraints during the operation, and sometimes due to a lack of interpretation experience.

Obviously, as a natural consequence, this paved the way for surgeons to wish for means to virtually assess the potential post-MVR-surgery outcome of specific rings for individual patients via a suitable biomechanical MVR surgery simulation. In this work, we will therefore be dealing with patient-specific simulation-based MVR surgery assistance.

3 Cardiac Surgery Assistance: State-of-the-Art and Current Open Challenges

In this chapter, in Section 3.1, we report on the *state-of-the-art* in cardiac surgery assistance. We look at the overall clinical information processing chain for surgery assistance – covering both the processing of imaging and non-imaging-related patient data – and finally focus on biomechanical simulation-based surgery assistance developments.

Second, in Section 3.2, we specifically discuss current *open challenges* and problems in the context of simulation-based cardiac surgery assistance.

This will lead us to the *concept* of future cognition-guided, simulation-enhanced surgery assistance, see Section 3.3. We motivate and detail the major research questions that have to be tackled in order for the development of a cognition-guided, patient-specific numerical simulation for cardiac surgery assistance. Concluding, we give a brief outlook on the *goals and structure of this work* and on the emerging cardiac surgery assistance system.

3.1 State-of-the-Art of Simulation-based Cardiac Surgery Assistance Workflows

In cardiac surgery, in order to make comprehensive diagnoses and decisions towards suitable surgery strategies and treatments, surgeons must account for large amounts of potential impact factors and relevant patient information: for instance, the MV morphology, which can be obtained from image data, or tissue properties and patient history records, which may be obtained from data that is acquired in other clinic departments [MSD⁺09, MVG⁺12].

In that regard, current IT developments gain increasing significance in today’s surgery assistance systems. This holds particularly when it comes to an intelligent and holistic processing of medical data [BBF⁺15], since surgically relevant information is usually not only stored in images but also in other data sources. Moreover, the information is typically distributedly available (if at all) on different computer and data management systems and has to be gathered and organized first.

Besides the huge amount of innovative IT developments in the context of medical information processing, there have been prominent advances in the development of prosthetic heart valves, and surgical valve repair techniques became increasingly effective, too. In the last two decades, heart valve surgery has therefore been characterized by major improvements, e.g., thanks to recent percutaneous devices like transcatheter AV implants (TAVIs), and MV annuloplasty devices [FFD⁺11].

Thinking further, it is apparent that this increasing sophistication of surgical solutions, the general availability of miscellaneous medical data, and the continuously broadening range of heart valve prostheses will require the development of patient-specific computer simulation

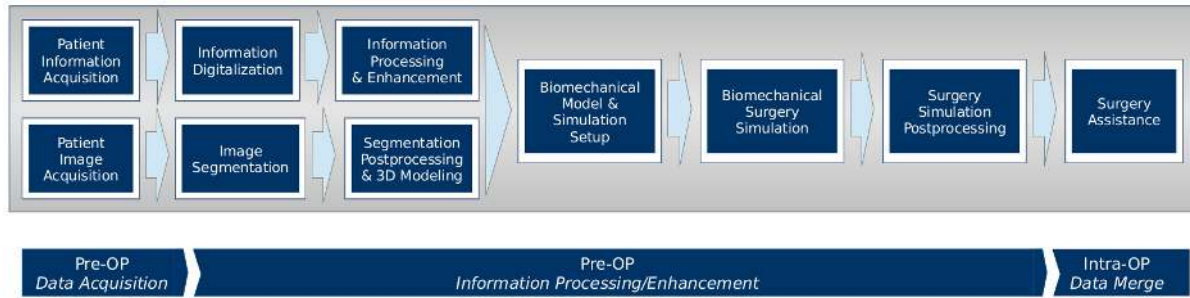


Figure 3.1: Scheme of the medical image and data acquisition and processing chain for simulation-based surgery assistance.

tools. These can support surgical planning and intervention through the assessment of preoperative scenarios and the prediction of post-operative and/or post-implantation outcomes. For the development of such tools, however, it is imperative that state-of-the-art medical imaging and data processing tools get integrated with clinical data management infrastructures and with biomechanical computational approaches [VLS⁺13, BBF⁺15].

In this section, we go through and report on the entire acquisition and processing chain of medical images and general patient data, eventually aiming for simulation-based surgery assistance, see Figure 3.1.

We start with the image processing chain, which goes from image acquisition via image segmentation and postprocessing, and continue with the general data processing chain which handles the broad range of medical patient information apart from images. Eventually, we will merge the resulting information and report on the state-of-the-art and advances in the context of biomechanical computer simulations for use in cardiac surgery support systems.

3.1.1 Image Acquisition

In the assessment of cardiac and particularly of valvular morphology and functionality, echocardiography, i.e., ultrasound imaging of the heart, has established as the imaging technique of choice: before, during and after heart surgery [LTH⁺10]. It is easy to use, portable, non-invasive and free of radiation, and operates at comparatively low cost. For the morphologic and functional analysis of the MV, real-time 3D transesophageal echocardiography (TEE) is extremely valuable, see again Figure 2.6, and recommended for guidance of interventional mitral valve procedures. Using 4D Color Doppler ultrasound imaging additionally allows for flow assessment, such that, e.g., mitral regurgitation becomes detectable. For more information on ultrasound imaging, we refer to [CP11].

Compared to other imaging modalities, the image quality of ultrasound imaging strongly varies and the viewable regions are limited by the surrounding structures and by the field of view of the ultrasound transducer. Moreover, temporal and spatial resolution mutually affect each other, and the respective dependencies have to be chosen carefully in the image recording protocols. In the assessment of valvular dimensions, for instance, multi-slice computer tomography (MSCT) and cardiac magnetic resonance imaging (CMR) have shown to provide higher

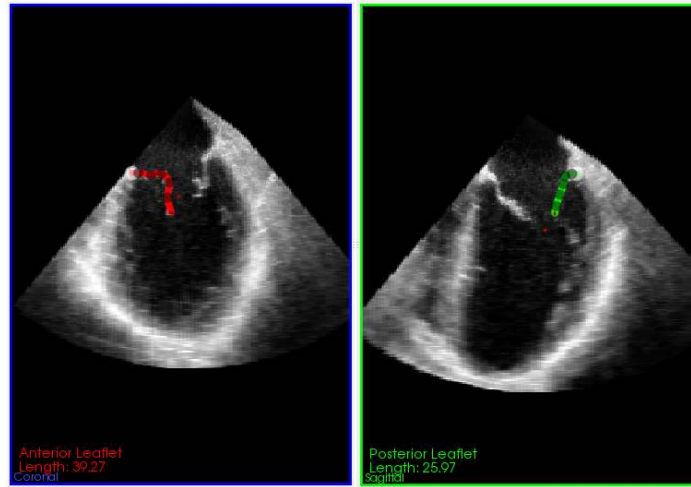


Figure 3.2: Color-coded representation of a 2D segmentation of the anterior (red) and posterior (green) MV leaflets in ultrasound images (more precisely: on 2D slices of ultrasound images). [Screenshot of a visualization obtained with the Medical Imaging Interaction Toolkit (MITK).]

reproducibility than echocardiography [JIM⁺11]. Hence, multimodality imaging is critical for future improvement of the accuracy of valvular measurements and for reducing the chance for prosthesis sizing errors.

3.1.2 Image Processing and Image Segmentation

Image segmentation constitutes the next big step in the whole surgery assistance chain of medical image and data acquisition and processing, see Figure 3.1. By a *segmentation of an image* we denote the contextual labeling of (somehow related) areas within this image. This can, for instance, be the contour of all pixels of the MV in an ultrasound image, see Figure 3.2.

Image segmentation is an important step in the context of surgery assistance, and a prerequisite for most subsequent steps in the (below) medical information processing chain, which if nothing else also contains the simulation step which this work focuses on. Manual segmentation, however, is very time-consuming, especially in 3D images, and hard to reproduce. Therefore, a variety of methods and toolkits have been developed to support, speed up and/or automate the segmentation process.

For a detailed introduction to the basics of image segmentation, we refer to [Jäh05]. Here, we only list the most popular *image segmentation approaches*:

- Simple region growing or thresholding methods, and more advanced diffusion methods;
- Deformable models [XPP00], in which pre-defined models are adapted to the image data;
- Active shape models or active contours [HM09], as well as statistical shape models [NMMH15], which make use of prior shape knowledge (so-called training data) in the learning-based segmentation process of specific objects.

Exploiting the specific features of these and other algorithms, there have recently been some promising developments in the context of valvular apparatus segmentation, which are likely to be usable for the quantitative evaluation of valvular pathologies: Schneider and colleagues [SPV⁺12] focus on the automatic detection of MV substructures. Jassar et al. [JBV⁺11] obtain 3D geometric reconstructions of the entire MV apparatus from ultrasound images, and Ionasec and colleagues [IVG⁺09] even allow for the dynamic modeling of the aorto-mitral coupling. Using automated learning-based algorithms on 4D CT images, Grbic et al. [GIV⁺12] proposed a model of the entire heart along with its four heart valves which is capable of estimating the global valvular location and motion. Engelhardt and colleagues [ELAM⁺15] present an automated approach for leaflet segmentation from 4D ultrasound images, which incorporates steps for modeling and computing the coaptation zone, and thus allows to assess the MV closing functionality from image data.

More recently, new ideas which integrate findings from probability theory and statistics also emerged in the research field of image segmentation: methods for Uncertainty Quantification seem very promising when it comes to the analysis of noise, artefacts and measurement inaccuracies in image data [MSW⁺15]. Also, Maier-Hein and colleagues [MHMK⁺14] investigated on the question whether masses of non-experts can train highly-accurate image classifiers, i.e., whether crowd intelligence can possibly replace segmentation efforts by clinical and radiological experts who are usually very cost-intensive and even more often limited in time.

Generally, applications for medical image processing require a solid underlying *software framework*, which not only allows software developers and medical imaging experts to design and develop new algorithms, but also provides an easy-to-use graphical user interface (GUI) to clinical users and staff. Here, we only want to mention the open-source *Medical Imaging Interaction Toolkit* (MITK), which is developed by the German Cancer Research Center in Heidelberg, Germany, and which provides these essential functionalities to visualize, process and interact with image data [NZS⁺13]. Resulting from and based on such a software toolkit, surgeons can obtain visualizations of 3D or 4D models of the valvular apparatus [ELAM⁺15], see, e.g., Figure 3.3. At the same point, these 3D or 4D models are the starting point for all ongoing work in the context of model- and simulation-based cardiac surgery assistance, see below.

3.1.3 Segmentation Postprocessing and 3D Modeling

Just until recently, image segmentation and the subsequent segmentation postprocessing represented the final step in the enhancement chain of medical image processing for surgery assistance, compare Figure 3.1. With the help of image segmentation, surgeons could already obtain a clearer 3-dimensional understanding of the morphologies of the objects that they were operating on. However, functional analysis was still up to their imagination and mostly based on (comparative) experience [VLS⁺13].

This obviously paved the way for surgeons and staff in the operation room (OR) to wish for patient-individual functional modeling and simulation in order to provide (virtual) insight into the operated organ and its behavior and functionality. To support pre- and intra-operative surgery planning and intervention, possible surgery scenarios could preoperatively be simulated, and post-operative surgical outcomes could virtually become assessable.

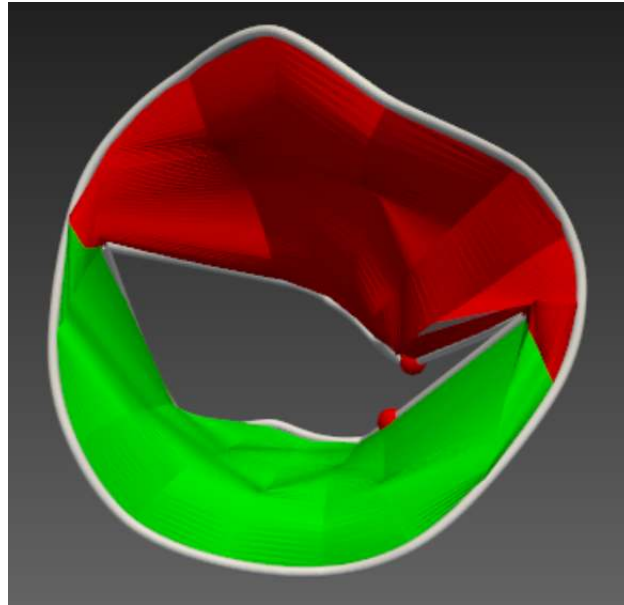


Figure 3.3: Color-coded representation of a 3D segmented model of the anterior (red) and posterior (green) MV leaflets, based on the above in Figure 3.2 shown 2D segmentation. [Screenshot of a visualization obtained with the Medical Imaging Interaction Toolkit (MITK).]

In a medical image and data processing pipeline which aims for such simulation-based surgery assistance, the next step after image segmentation is hence usually constituted by the 3D modeling of the object of interest, in our case of the MV, see Figure 3.3.

Manual modeling of the MV based on image data can be very time-consuming, and for everyday in-OR-application automated or semi-automated modeling is imperative. However, manual modeling by an expert still is the most precise and reliable method and to date provides gold standards for the evaluation of automated and semi-automated modeling approaches.

In 2012, Pouch and colleagues [PXY⁺12] proposed a semi-automated approach for MV modeling based on 3D ultrasound images. After manual selection of an initial region of interest which contains the MV and of a set of points on the MV annulus structure, the algorithm automatically extracts a 3D region of interest which only contains the MV. The subsequent execution of a thresholding and active contour evolution yields a 3D geometrical model of the MV leaflets. Ionasec et al. [IVG⁺10] developed a method for automated 4D modeling of AV and MV from CT and TEE images. Its coarse-to-fine machine learning approach starts finding the global location and rigid motion of the valves on a 4D image. Following a non-rigid landmark motion estimation is performed by trajectory spectrum learning, and finally combined with a shape and manifold motion model method. It thus allows for obtaining all important anatomical substructures of the MV fully-automatically, however, the method requires a broad set of training data, which necessitates long setup times for specific anatomies. Ionasec's initial algorithm was extended by Voigt and colleagues [VMI⁺11, VIG⁺09], who included biomechanical constraints into their automated geometrical MV segmentation and modeling process to

improve the algorithm’s accuracy and decrease its susceptibility to errors. However, these constraints were mainly dominated by the respective MV tissue properties (reported by Kunzelmann et al. [KCC⁺93]) and do not yet include prior knowledge of, e.g., shape and curvature of an MV.

Of course, still before entering the field of simulations, these 3D geometrical models of the MV allow for various *quantitative* surgically-relevant assessments. For instance, Engelhardt and colleagues proposed a method for automatic MV coaptation zone assessment [ELAM⁺15]. Graser et al. developed a system for the robust modeling of the mitral annulus on 4D TEE image data which is based on MV annulus shape priors, and which yields additional information for patient-specific implant planning [GWAM⁺14]. Yet, as explained above, functional analysis and simulation of the patient-individual organ and tissue behavior was still out of reach, just as any knowledge-based surgery simulation approach.

3.1.4 General Information Processing, Knowledge Representation and Data Science in Surgery Assistance Environments

For *holistic* surgical treatment planning one cannot restrict medical information processing to the processing of images only, but typically must also account for all other types of data and information that is recorded in the context of the respective patient’s medical and clinical treatment [MSD⁺09]. This data includes, for example, the patient’s personal information (age, sex, etc.), health history records (on medication, previous diagnoses, treatments and surgeries, etc.), medical and clinical parameters (from laboratories, radiology, anaesthesia, etc.), and generally everything else that is recorded before, during or after the respective surgical treatment.

In Figure 3.1, we depict this ‘other’ type of information, i.e., the data which does not consist of (raw or processed) medical images, by means of the upper left three boxes in the merging information processing chain. Together with the information consisting of medical images, the upper and lower left rows jointly represent the entire set of surgically relevant pieces of information and the corresponding clinical and surgical information processing pipeline, up to the point where a possible surgery simulation setup and hence a simulation-enhanced surgery support workflow may be started.

It is important to emphasize at this point that the respective information is acquired and gathered in possibly *independent* clinical departments and hence in *separated* clinical data management infrastructures [FMK⁺16]. Also, still today, the bigger part of patient data is *not digitally* available, since clinical staff is still commonly working on the basis of handwritten notes and pre-printed forms [Ber06]. The data hence often still needs to be *digitized*, and the contained information must be *semantified*, i.e., semantically annotated, before any reasonable machine-based information processing is enabled at all [FMK⁺16].

However, besides the data itself, in order for a machine (e.g., a surgery assistance system) to adequately process the data, it is imperative for the system to have access to a machine-usable representation of surgical knowledge and experience. This surgical knowledge and experience is what human surgeons commonly acquire over the years: *factual knowledge* from literature, scientific or medical/clinical studies, guidelines, etc., and *practical knowledge*, i.e., experience and expertise in surgical procedures, etc. A machine-interpretable representation of this knowl-

edge is what actually will allow a surgery assistance system to properly analyze, process and interpret the given data, and to thereon-based possibly deduce an optimal surgical treatment strategy [FMK⁺16].

Fitting these requirements, and going hand in hand with the fast progress in the field of Data Science (DS), Machine Learning (ML) and Artificial Intelligence (AI), recent IT developments gain increasing significance in today's surgery assistance systems, especially when it comes to intelligent and holistic analysis and processing of medical data and information [BBF⁺15].

Above all, *knowledge representation and reasoning*, a field of AI which aims at representing information about the world in a form that a computer system can utilize in order to solve sophisticated tasks, has gained importance, e.g., in the diagnosis of medical conditions or in the selection of surgical treatment strategies. Several formalisms such as semantic nets or ontologies have been conceptualized and implemented in order to drive knowledge representation, e.g., for the application in expert systems. Thereon-based, an impressive number of automated reasoning engines (e.g., inference engines, theorem provers, classifiers) have been developed and are today embedded or integrated in a range of intelligent devices and systems to thus help automate everyday life, not only in the surgical domain.

As of 2001, projects associated with the *Semantic Web* represent one of the most active research fields in the area of knowledge representation and reasoning [BLHL01]. The Semantic Web extends the Web by means of a layer of semantics on top of the current Internet layer, in which information is given a well-defined meaning to thus allow machines to read, interpret and process the data and knowledge using inference and deductive reasoning techniques. In order to facilitate this idea, the Semantic Web builds on the principles of *Linked Data* – according to Tim Berners-Lee – for publishing structured data so that it can be interlinked and become more useful through semantic queries [BL06]. Also, through utilization of the common *RDF data model*, which is based on triples consisting of subject, predicate and object, and a standardized *ontology language*, data and even implicit knowledge worldwide can be integrated [OGS09]. Ontologies are used for the formal declaration (naming and definition) of things, types, properties, attributes and relations between things and entities of a particular domain of discourse. They thus allow to limit complexity and to organize information, which eventually is essential to capture and manage knowledge [OGS09]. To give ontologies a substantial formal basis, large joint projects have recently integrated concepts from knowledge representation and reasoning (namely: frame languages and automated reasoning components, so-called classifiers) with markup languages based on XML. Thus, the Resource Description Framework (RDF) provides basic capabilities to define knowledge-based objects by means of basic features such as subsumption relations and object properties. In addition to that, the Web Ontology Language (OWL) adds levels of semantics and hence enables integration with classification engines [BLHL01].

Driven through rapid progress and ongoing vivid research and development activities in the above fields, and thanks to the intelligent concepts and techniques which thus were in reach, recent years have shown a number of sophisticated and highly-specific advances thrive in the domain of biomedical informatics, and in particular in surgery assistance.

With respect to the management and handling of data, it is again to be emphasized that seemingly simple tasks – as the above mentioned data digitalization and semantification – have already posed big challenges, that had to be dealt with before any reasonable machine-based

information processing techniques could actually be addressed [FMK⁺16]. Moreover, the data is usually only accessible through a number of highly specialized clinic information systems (CIS) and proprietary user interfaces, which constitutes an additional challenge.

This being said, several research groups around the world have made various efforts in order to develop systems which enable the huge amount of *heterogeneous* and *distributed* data to be accessed, processed, interpreted and incorporated, such that the therefrom resulting diversity of clinically relevant parameters could properly be extracted and further processed, e.g., for surgery support purposes. In the beginnings, a number of simple tools for data digitalization and annotation or semantification came up, only few yet relying on the above mentioned RDF data model and the OWL ontology language. Tools and initiatives like the prevailing Extensible Neuroimaging Archive Toolkit (XNAT) [MORB07] or the Collaborative Informatics and Neuroimaging Suite (COINS) [SCW⁺11] then focused on storing and sharing data to support multi-institutional research. Building up on this, the i2b2 project (informatics for integrating biology and the bedside) [MWM⁺10] aimed at the facilitation of comprehensive statistics on such data in order to deduce new insights. Going one step further, Chen et al. [CHBC12] suggested a medical recommendation system based on domain ontologies and the rule language SWRL to select anti-diabetic drugs. Similarly, the utilization of ontologies for describing the medical concepts of different source and target systems enabled the integration of data between clinical and research systems [MKT⁺15]. Following Panahiazar et al. [PTJP14], adding a semantic meaning to data from heterogeneous sources can facilitate the creation of clinically relevant knowledge in order to eventually enable a more personalized treatment.

Beyond this, we want to list a few important advances in the context of knowledge representation systems and information processing architectures in detail:

On the one hand, Gemmeke et al. [GMP⁺14] implemented an information architecture that is based on Linked Data and Web APIs in order for the automation of the preprocessing of medical images and to thus *support data consolidation and integration*. Their solution enables the flexible composition and execution of clinical data processing pipelines by the example of brain tumor progression maps. The individual (possibly distributed) processing steps are exposed through semantically described Web APIs and the respective input and output data are specified via their unique resource identifiers (URIs). They appraise their architecture to be transferable from image related data processing pipelines to arbitrary chains of individual algorithms.

However, there has not been a general concept for integrating the heterogeneous patient data that accumulates throughout the *entire surgical pathway* with scientific data that is derived from research or published resources until 2015. Nor is there a proposal for using a thus obtained integration for setting up a context-aware surgery assistance system that provides the surgeon when required with the available information through comprehensive information processing, thus enabling improved surgical treatment planning [FMK⁺16].

On the other hand, going towards the latter direction, in 2015, März et al. [MHW⁺15] proposed a *concept* which enables joint storing and processing of heterogeneous data from various information sources aiming for the facilitation of a holistic surgical treatment strategy planning for liver surgery. They figured that there are three general sources of information which influence a human surgeon's surgical strategy planning: *patient-individual data*, as described above, including images and parameter data, *factual knowledge*, which is written down in quotable

sources like clinical guidelines, studies, or educational books, and *practical knowledge*, which results from surgical experience. In their concept, they accordingly represented these information sources: they formalized a *dynamic patient model* that incorporates the heterogeneous data acquired for a specific patient in the whole course of disease treatment, and designed a *surgical reasoning system* which maps and accordingly uses the factual and practical knowledge. All this is to be built up on a technical infrastructure [FMK⁺16] that enables the storage of, access to, and modeling and processing of an arbitrary variety of heterogeneous data to eventually support clinical decision making. Yet, to date, clinical studies, an evaluation, and a general proof-of-concept are missing; not to mention that there is no evidence for the long-term beneficial effects which this conceptual setup might really have.

Lastly, a set of very important developments in the area of knowledge representation and intelligent information processing, has been achieved by the company *IBM* in the context of the *Watson projects*, which combine methods and concepts from Data Science with advances in Artificial Intelligence in order to – among others – bring forth cognition in surgery assistance systems and foster knowledge management in clinic information systems. Watson is a general question answering (QA) computing system, developed in IBM’s DeepQA project, which is built to apply advanced natural language processing, information retrieval, knowledge representation, automated reasoning, and machine learning technologies to the field of open domain question answering [Mur15]. In healthcare, Watson’s natural language, hypothesis generation, and evidence-based learning capabilities allow it to function as a clinical decision support (CDS) system for use by medical professionals [GMSM13]. In particular, IBM Medical Sieve is an ambitious long-term exploratory project which aims at building a next generation *cognitive assistant*, which is supposed to exhibit a deep understanding of diseases and their interpretation, with advanced multimodal analytics, clinical knowledge and reasoning capabilities, such that it is qualified to assist in clinical decision making in radiology and cardiology [TKT⁺13].

In their review and position paper [BBF⁺15], IBM report on their comprehensive understanding of the state-of-the-art in what they call research and development towards the *cognitive automation of data science*, with a particular focus on medical and clinical applications. They describe the perpetual potential of Machine Learning and Data Science approaches to process and leverage heterogeneous medical information (i.e., images and other types of data) and to infer new medically and clinically relevant knowledge.

Yet, it becomes apparent that, so far, none of these approaches considers and plans for the integration of *biomechanical surgery simulations* into their comprehensive surgery treatment planning systems. However, in this context, it is obvious that there are still two major future required improvement areas before a system as the proposed one can possibly *go live*: 1) an in-depth evaluation and backup of the numerics, the modeling and of the simulation system setup is needed, far beyond a prototypic, single-feature-focused implementation, as well as an automated personalized model/simulation calibration, and 2) for integration purposes and in order to share ideas within a hopefully growing community, semantic representations must become more generalizing, connecting and conforming with other semantic annotations and ontologies, and must additionally allow to fully control the system execution.

Summarizing, the combination of great recent advances in the area of Artificial Intelligence and Data Science (especially with respect to formal description methods, as well as in what concerns analytical and reasoning techniques) with the general availability of large amounts

of heterogeneous patient data (both digitized or analogue, structured or unstructured) and surgical knowledge (both based on experience or on scientific research) seems very promising [SVIH16]. Surgical Data Science appears to afford an opportunity to address difficult challenges in ensuring that surgery (and healthcare in general) is safe, effective, efficient, patient-centered, equitable, and timely. For more details and an in-depth analysis of the perspectives, challenges, scope and objectives of surgical data science, we refer to [SVIH16].

3.1.5 Biomechanical Modeling and Surgery Simulation

Looking at the surgical information processing chain in Figure 3.1, the next big step we consider comes after *merging* the above described information and is constituted by the functional modeling and numerical simulation of the soft tissue behavior.

Thus, combining the results of image processing (i.e., in our case, the geometrical 3D models of organs or organ parts) with the surgical knowledge and the derived constraints (i.e., in our case, e.g., the material properties or the implantation of a ring prosthesis) and adding on top the knowledge and the equations from continuum mechanics, one can obtain a description of the behavior of the respective tissues subject to internal or external forces and momentums. This description, the *functional modeling* of the behavior of soft tissue, is the starting point for *numerical surgery simulations*. It preferably accounts for all relevant pieces of information in the above medical data processing chain, see Sections 3.1.1 to 3.1.4, and thus allows to compute and virtually reproduce the effects of forces and momentums, which occur, for instance, during or after surgical manipulation. This opens up great new opportunities for surgery assistance and training, and has a great potential to improve diagnosis and therapy: Surgeons in the OR are interested, e.g., in the tissue's stress and deformation behavior, and in the MV leaflet coaptation capability, both under natural and surgically manipulated conditions. Using patient-individual simulation-based surgery assistance, they may be provided preoperatively with visualizations or quantitative analyses of their patients and of their possible post-operative surgical outcomes, e.g., after an annuloplasty ring implantation. Also, the effects of different surgical manipulation techniques (e.g., virtual suture, leaflet resection, or subvalvular repairs) on geometry and stress could be virtually examined prior to studies on animals and humans [SVO⁺15].

In a *comprehensive* surgery assistance system, a feature which is capable of the functional modeling and the numerical simulation of the soft tissue behavior is hence practically imperative, and enormously fosters the understanding of complex, biomechanical, functional characteristics of the tissues.

Principally, one can classify four essential components which every surgery simulation system must contain in some form; compare Figure 3.1:

1. A **numerical modeling and simulation component**, see subsection 3.1.5.1, the central component of simulation-enhanced surgery assistance. It defines the properties of the underlying biomechanical model and transforms these into a mathematical problem formulation (thereby considering, e.g., the body's morphology, the elasticity behavior, boundary conditions, etc.). On the basis of this model description it allows, e.g., via using the Finite Element Method (FEM) to execute the respective simulation scenarios as obtained from the simulation preprocessing component.

2. A **simulation preprocessing component**, see subsection 3.1.5.2, supports the setup of the above described simulation scenarios by analyzing and transforming simulation input data (which comes from the information processing steps described in Sections 3.1.1 to 3.1.4) into data structures which are conforming with the biomechanical model and which are suitable for the simulation application. Preferably, the simulation preprocessing step is controlled by some sort of intelligent, cognitive system in order to (possibly fully-automatically) set up situation-adapted, patient-specific surgery simulation scenarios.
3. A **simulation postprocessing component**, see subsection 3.1.5.3, which transforms the simulation results in a way such that they are easily understandable for surgeons, e.g., by means of visualization of the MV morphology throughout the cardiac cycle on a 3D screen. Also, it may provide the surgeon with pre-defined surgically relevant quantities, such as critical distributions of tissue tensions due to surgical manipulation, or the virtual post-operation size of the coaptation zone.
4. A **simulation interface component**, see subsection 3.1.5.4, finally enables the seamless integration of the surgery simulation application and its pre- and postprocessing components and/or results into a surgery assistance workflow and assistance system.

We will have a closer look at the state-of-the-art with respect to all of these components in the following.

3.1.5.1 Numerical Modeling and Surgery Simulation

In order to solve the equations of continuum mechanics, which describe the behavior of elastic soft tissues, there are various numerical methodologies. They all allow to represent the biomechanical tissue behavior and at some level its coupled interaction with the implanted devices and with the blood flow [VLS⁺13]. Depending on these methodologies' degree of sophistication and on their capability to capture and reproduce the diverse peculiarities of the anatomic behavior, the underlying models and the resulting simulations can yield a spatial and temporal variability of a range of clinically relevant quantities: e.g., tissue deformations, mechanical stresses in the tissues, blood fluid-caused wall shear stresses, regions of blood stasis or turbulence, etc. Generally, one can categorize these methodologies in three principal classes:

- Mass-Spring Model (MSMs) simulations: These describe a continuous structure (i.e., a body or organ) as a cloud of discrete mass points that are connected by a network of springs to compute the response of the structure to mechanical stimuli. In spite of their considerable potential as real-time patient-specific simulation tools, the application of MSMs to heart valve biomechanics is still in an embryonic stage, and only a few works have very recently made contributions in this context [HSdNH12, THF⁺13, SSS⁺15].
- Structural Finite Element Method (FEM) simulations: In the FEM, the structural system is modeled by a set of appropriate finite elements that are interconnected at points called nodes. These elements have physical properties such as thickness, density, shear modulus or Poisson's ratio, and tissue field variables such as displacements, stresses and strains are then computed on the basis of boundary value problems for partial differential

equations. Typically, it is assumed that the effect of fluids (e.g., blood) can be represented simplistically through prescribed pressures. We will give a more detailed overview on FEM-based simulations just below.

- Fully-coupled Fluid Structure Interaction (FSI) simulations: The underlying models explicitly account for the coupling between the valve tissue and the surrounding blood flow, and thus overcome this inherent limitation of the classic FEM. Here, also, we will give more details on FSI simulations in the context of cardiac surgery assistance just below.

In the last 20 years, FEM and FSI models and thereon-based simulations have been increasingly adopted to study the biomechanics of natural and prosthetic valves [VLS⁺13]. FEM-based simulations are computationally more efficient than FSI-based simulations, as they can handle the complexity of anatomic geometries by decoupling the fluid and structural domains whereas FSI methods on purpose do not. For that reason, FEM models have been established to provide powerful insights into clinically relevant patient-specific situations, see, e.g., the works of Stevanella et al. [SMC⁺11] or Mansi et al. [MVG⁺12]. In contrast, FSI models are computationally a lot more expensive, especially in 3D, and their application has therefore long been restricted to the study of valves in simplified domains, such as in the work of Griffith [Gri12] who employs an immersed boundary model to straight axisymmetric aortic lumens. More recently, attempts in more complex anatomic, patient-specific domains have began to emerge. Le and Sotiropoulos [LS13] have investigated on the coupled interaction of valves and their complex hemodynamic environment in the left ventricle. Similarly, Gao et al. [GMQ⁺14] have set up an FSI MV model which was manually constructed on the basis of a human volunteer's MRI images. Despite minor discrepancies, comparative validation with the same volunteer showed an overall good agreement of model and reality for both the opening and the closing MV configuration, and motivated the promising application of FSI-based simulation for mimicking the in vivo MV dynamics. In the same context, Sotiropoulos [Sot12] figured and predicted that with the rise of massively parallel computational platforms and with continuous algorithmic advances – not only in what concerns the simulation algorithms themselves, but also with respect to simulation preprocessing –, 3D FSI simulations of cardiac valves in patient-specific left heart anatomies are well within reach. Accordingly, Marom [Mar14] summarizes great recent achievements in that context and reports on the state-of-the-art in 2014 with a particular focus on FSI models for the aortic valve. He clearly states that during recent years FSI models have shed light on phenomena that could not be captured in experiments or in simplified models of solely hemodynamics or structural mechanics.

Looking at the diversity of works and advances in this context, we refer to three important recent reviews, which provide a great overview: In 2013, Votta and colleagues [VLS⁺13] summarize topical works towards patient-specific cardiac valve simulations. Chandran and Kim [CK14] report in 2014 on FEM- and FSI-based computational MV evaluation and potential clinical applications, therein emphasizing the dependence on high-quality imaging techniques. Early in 2016, Ratcliffe et al. [MPW⁺16] conclude with a description of the evolution of FEM-based models and simulations of mitral valve repair in particular.

We summarize that for a computational model to evolve into a clinically relevant simulation tool it must be able to capture the following four principal features, for which we respectively list important results and the some prominent and advanced works:

- **The heart and valve morphology:** In order to properly reproduce the heart and valve anatomy and functionality as described in Section 2.1, all four principal substructures (mitral annulus, anterior/posterior leaflets, chordae tendineae, and papillary muscles) have to be included in the biomechanical model. Regarding the automation of capturing these substructures, the gold standard is likely represented by the works of Pouch et al. [PXY⁺12] and Mansi et al. [MVG⁺12]: They obtain detailed 4D representations of the annulus profile, papillary muscle tips, and leaflets; based on a Machine Learning approach, Mansi and colleagues were able to even capture the leaflet thickness distribution, which was shown to strongly affect the leaflets' stress patterns. With respect to the chordae distribution, only Wang and Sun [WS12] had been able to patient-specifically describe the chordae origins on the papillary muscles and their respective leaflet insertions. However, no 4D profiles could be obtained, which, however, would be required in order to properly capture their restricting effects with respect to leaflet closure and prolapse, and which represent essential kinematic constraints [PHS09].
- **Mechanical tissue properties:** A fully patient-specific model needs to cover patient-specific mechanical tissue properties. MV tissues are soft and strongly hydrated, and when tested *ex-vivo* they undergo large strains and show a heterogeneous, incompressible, non-linear and anisotropic mechanical behavior [SKV⁺11]. However, *in-vivo* analysis of human MV tissue is very challenging and has rarely been done so far. Lee et al. [LAG⁺13] and Kanik et al. [KMV⁺13] have proposed frameworks to estimate patient-specific MV material properties using 4D TEE ultrasound imaging. They combine image-derived MV dynamics with a biomechanical model to estimate regional patient-specific material parameters *in vivo*, and thus point the way towards the right direction. However, further work is needed to improve the prediction/correction terms of their inverse FE iterations with respect to resolution and reliability. More in-depth analysis of *in-vivo* MV tissue properties was indirectly conducted on ovine models through inverse FEM by Krishnamurthy and colleagues [KIS⁺09]. They found that the anterior leaflet has an almost linear and anisotropic material response, with the elastic modulus in the two principal directions of the leaflet one order of magnitude higher than those measured *ex-vivo*, and they showed that this is due to the presence of neurally-controlled active contractile elements in the MV leaflet tissues [SKI⁺11, PS09]. In spite of these findings, whose implications may be transferred to human MV tissue (see, e.g., the work of Wang and Sun [WS12], or of Kunzelman et al. [KCVE94]), current patient-specific MV models account for material properties that are based on *ex-vivo* tests with porcine. These suggest to model the MV leaflet response to be either *linear orthotropic elastic* or *transversely isotropic hyperelastic* [KCVE94]. The chordae are assumed to be linear elastic or isotropic hyperelastic [WS12, PHS09]. Yet, while the tissue properties may change notably between healthy and pathological conditions, and between different pathological states (e.g. functional mitral regurgitation vs. myxomatous disease), within a well-defined clinical scenario the inter-subject variations of mechanical properties were shown not be as relevant as those in morphological features of the valve and its environment [PSSH10]. Therefore, classifying tissue properties according to such pre-defined clinical scenarios seems reasonable.

- **Kinematic constraints:** As explained in Section 2.1, dimension and shape of the left ventricle vary dynamically during the cardiac cycle. The ventricle motion entails that both MV annulus and papillary muscle tips are in motion, too, and change position while contracting and relaxing, thus regulating chordae tension. Also, according to Section 2.2, the implantation of an annuloplasty ring prosthesis onto the natural annulus changes its original size and shape. In patient-specific biomechanical models, these dynamic changes must be accounted for, both in a natural and in a surgically manipulated case, by means of kinematic constraints [PHS09]. Several approaches have thus worked towards tracking the in-vivo motion of ventricle and valve in order to have the respective virtual motion imposed by means of kinematic boundary conditions, or in other words through time-dependent nodal displacements. For instance, Stevanella et al. [SKV⁺11] used 4D Cardiac Magnetic Resonance (CMR) long-axis cut-planes to track the papillary muscles' motion in order to exemplarily conduct manually set up patient-specific FEM simulations. Their results suggest the general feasibility and importance of this approach, however, their work also highlights further needed developments to satisfy the requirements for clinical application, namely flexibility, generalization, usability and time-efficiency. Similarly, Mansi et al. [MVG⁺12] successfully employed machine learning algorithms on 3D real-time TEE ultrasound images in order to capture the valve motion for integrated FEM-based MV simulations. Modeling the surgical procedure of an MVR, i.e., the implantation of an annuloplasty ring, the major pioneering work comes from Votta et al. [VMB⁺07], who investigated by means of an FE study on the effects of two annuloplasty rings with respect to the virtual post-implantation stress distribution and closure behavior. However, their study was conducted on a generalized MV model and not on patient-specific 3D data. Recent advances were made, e.g., by Wong et al. [WWZ⁺12], who used different types of annuloplasty rings along with an ovine MV model in order to investigate the differing effects of saddle-shaped versus asymmetric ring prostheses. They found that all rings were – when properly implanted – capable of restoring normal valvular geometry and of eliminating mitral regurgitation. Morrel et al. [MGZ⁺14] translated this work to human MV models, and investigated on leaflet coaptation as well as on valvular, annular and suture stress distributions for three different annuloplasty band shapes. Their finding states that all rings increased leaflet coaptation and reduced stress across the MV leaflets, however, their results suggest that saddle-shaped prostheses were the most effective, as were smaller sizes across all ring shapes. However, to date, there is no framework to *automatically* process patient-individual heart and valve models along with appropriate kinematic constraints in combination with suitably chosen annuloplasty rings in order to obtain patient-specific FEM-based annuloplasty simulations for surgery assistance. In this context, to the best of the author's knowledge, the pioneering work on the integration of a patient-specific biomechanical annuloplasty surgery simulation into a knowledge-based surgery assistance system was presented in 2015 by Schoch et al. [SEZ⁺15]. Dedicated works on the implementation, the employment and specific system components followed in 2016 [SPW⁺16, FMK⁺16, SKS⁺16].
- **Heart-valve-blood interaction:** In FEM models and simulations, the treatment of blood flow and heart-valve-blood interaction is usually simplified via prescribed pressures. Thus, accounting for the finding that the MV opening and closure are mainly driven by

the physiological transvalvular pressure drop (of normally up to 120 mmHg), in structural FEM simulations the effect of blood is hence modeled by applying pressure loading on the ventricular side of the leaflets for MV closure and on the leaflets' atrial side for MV opening, see, e.g., [SKV⁺11]. It has been shown that, in general, the MV function can be realistically captured through this approach [WZC⁺10]. However, the local features of blood flow can play a non-negligible role in specific applications such as for the analysis of the left ventricle filling process through the MV [LDSB10, VCV⁺08]. This however is not in the focus of an MVR surgery assistance development.

Concluding, there have been many great works and a remarkable overall progress in the modeling and simulation of the MV behavior. Different research groups and institutions worked on different, partly highly-specific and restricted problems in order to appropriately model and simulate specific physiological features, such as the chordal effects or the integration of particular surgical manipulation procedures. An all-encompassing work, however, which incorporates, involves and represents all of these advances is still missing, and patient-specific, situation-adapted, realistic FEM simulations still commonly need to be set-up and calibrated manually.

Regarding the simulation software, one finds that there is a number of open-source physics and simulation engines available, the most prominent ones of which are SOFA [FDD⁺12], HiFlow³ [AAB⁺12], DealII [BHH⁺16], or Dune [DKNO10]. Besides that, there are advanced commercial simulation software packages available for purchase on the market; among these are Abaqus, Ansys, COMSOL Multiphysics, FEBio [MEAW12], Matlab, Materialise Mimics and 3matics.

3.1.5.2 Simulation Preprocessing

In order to obtain meaningful and beneficial surgery simulation results, FEM simulation scenarios and their underlying biomechanical models should account for all patient-individual data and consider surgical expert knowledge to appropriately represent the surgical manipulation. At the same time, the simulation algorithms themselves need to be optimally tuned with respect to the underlying numerics and their efficiency.

The simulation preprocessing component is to support the problem-specific *selection* of the respective numerical properties and the *setup* of patient-specific simulation scenarios: It therefore analyzes and processes relevant simulation input data which comes from the information processing steps described above in Sections 3.1.1 to 3.1.4, and transforms it into data structures which are conforming with the biomechanical model and which are suitable for the respective simulation application, see Section 3.1.5.1. Also, in order for an efficient and realistic simulation, a lot of expert knowledge must be applied in the numerical simulation setup, e.g., with respect to the choice of appropriate modeling techniques, material models and competent solvers for the respective problems.

Usually, the data generation in the before-mentioned steps 3.1.1 to 3.1.4 involves a number of different tools, such as imaging softwares, segmentation and meshing algorithms, etc. Also, lots of different data types and formats are obtained in the data processing chain.

The entire workflow, at the end of which one can construct and obtain patient-specific biomechanical models from tomographic and parametric patient data, can – on an abstract

level – be divided into five major steps: Firstly, after the relevant anatomical structures have been segmented from imaging data, a surface mesh is constructed, and possibly, different surface mesh enhancement operators are applied, e.g., for smoothing or coarsening. Secondly, from the surface mesh one can construct the volumetric computational grid, which is to represent the FE body region. Additional material properties and boundary conditions then complete and define the mathematical problem, such that subsequently, a simulation engine can be set-up, e.g., by choosing appropriate efficient solvers, in order to then perform the simulation. Finally, the simulation output is post-processed for surgically relevant quantifications, 3D visualization or diagnostic purposes, see Section 3.1.5.4.

Please note, that this holds both for the specific case of cardiac surgery assistance and for general medical image processing for simulation-based surgery assistance.

For each of the above mentioned steps, there exists a diverse ecosystem of software tools and algorithms. There are many established *open-source packages and toolkits* for

- segmentation (3D Slicer [PHK04], ImageJ [SRHE15], MedINRIA [TSF07], MITK [NZS⁺13]),
- surface and volumetric mesh processing / meshing (CGAL [FP09], Meshlab [CCC⁺08], Tetgen [Si15]),
- biomechanical simulation (Deal II [BHH⁺16], Dune [DKNO10], FEBio [MEAW12], HiFlow³ [AAB⁺12], SOFA [FDD⁺12]), and
- post processing (ParaView [HAL04]).

Apart from these open-source toolkits, biomechanics applications also frequently make use of *commercial CAE packages*, of which the most established ones are ABAQUS, Ansys, FEBio, Matlab, Materialise Mimics and 3matic. Finally, many research institutions develop their own highly problem-specific tools (for instance, those mentioned in Section 3.1.5.1) in order to optimize or automate a single step in the simulation workflow or to reduce the run-time.

In order for the setup of problem-specific data processing chains – and, thus, also in order for the simulation preprocessing for simulation-based cardiac surgery assistance – different tools need to be combined and connected. However, this seemingly simple task is often very time-consuming, since there is a severe lack of common standards with respect to software interfaces and data formats.

For instance, the data formats for surface meshes are at least compatible in large parts (e.g., via the `.stl` format), but there are numerous formats for volumetric meshes (`.msh`, `.vtk`, or `.gsh`, to just name a few) and many tools are incompatible as they do not share a common format. This is particularly complicated when it comes to describing material properties, defining boundary conditions or setting simulation parameters, since seemingly every simulation toolkit has their own format, which enormously hampers prototyping work and research.

Several institutions and initiatives have made a number of attempts to overcome this problem, e.g., by describing and connecting the different toolkits via a common interface. For instance, the Virtual Reality Modeling Language (VRML) [CB97] is one of the first proposed

languages that enables general XML-based descriptions for physical objects. The Medical Reality Modeling Language (MRML) [GNK⁺01] allows for an XML-based representation of data that are typically arising in image-guided surgery, and handles the most common medical image data types. When it comes to describing surgical workflows and organ models for surgery training, the Surgical Simulation and Training Markup Language (SSTML) [BTP⁺06] has been well-established over the recent years. Concerning biomechanical soft tissue simulations, the Physical Model Language (PML) [CP04] was designed to allow for a unified formulation of both finite element and mass-spring models. Similarly, other general-purpose simulation softwares, toolkits and libraries, such as SOFA, HiFlow³, DealII or Dune, use elaborate XML-based formats, too, in order to describe their proprietary biomechanical model and simulation setups; see, e.g., [FDD⁺12, AAB⁺12].

However, to the best of the author's knowledge, until 2012, there has not been any model description language that is capable of representing not only the entire biomechanical modeling workflow but also the biomechanical model and simulation scenario setup, that can then be used and performed by the simulation engine [SSS⁺14]. As a direct consequence, this often entailed upcoming problems with incompatibilities between model descriptions and simulation engines used. Moreover, extensions of existing model descriptions (e.g., towards a new material model) and the integration of new components (e.g., parallel solvers, or penalty terms to account for contact) into specific modeling pipelines required an in-depth knowledge of the respective modeling language and the specific framework. This is why, in the framework of the transregional collaborative research center *Cognition-Guided Surgery* (SFB / TRR 125), we proposed the Medical Simulation Markup Language (MSML) [SSS⁺14] to overcome this problem. The MSML scheme describes a biomechanical model and the according simulation scenario including all relevant information, such as mesh properties, material models and boundary conditions. Its unified description is parsed/interpreted and can be exported to different simulation engines and applications. Moreover, at the same time, as a unique feature of the MSML, it is able to also describe all steps in the modeling and simulation workflow, and can thus act as a middleware between all tools used in the modeling pipeline.

Looking at the specific case of simulation preprocessing for patient-specific simulation-based cardiac surgery assistance, we recall that there are many approaches which successfully preprocess subsets of pieces of information for very specific focuses that are given in the respective simulation applications; please see again the works mentioned above in Section 3.1.5.1.

To just name a few examples, Pouch et al. [PXY⁺12] and Mansi et al. [MVG⁺12] automatically obtain detailed 4D representations of annulus profile, papillary muscle tips, and leaflets, yet the chordae distribution and a comprehensive automated setup of appropriate boundary conditions was missing. Similarly, even though first pioneering works with respect to the consideration of kinematic constraints were made, e.g., by Votta et al. [VMB⁺07] or by Morrel and colleagues [MGZ⁺14], to date, there is no common framework or standard to automatically process this data and information along with patient-individual heart and valve models in order to set up patient-specific biomechanical simulations. Just until recently, there had not been any successful attempts to analyze in a simulation preprocessing step the 3D heart/valve morphology in order to extend a biomechanical model with patient-individually chosen annuloplasty rings (and the respective additional boundary conditions). Choi et al. [CRMK14] now do so

in order to obtain patient-specific FEM-based annuloplasty simulations for surgery assistance, however, they only consider imaging and 3D morphology information, and do not yet account for other relevant patient-specific parameters, such as the patient’s health history, etc. Also, whilst other current works provide partly manual, partly semi-automatic, separate building blocks to accomplish the simulation preprocessing task of setting up patient-individual simulation scenarios, there has not been a framework to comprehensively and intelligently combine and automate them [SKS⁺16]. Schoch et al. for the first time proposed such a system which is capable of the latter in 2016 [SPW⁺16, SKS⁺16].

Summarizing, what was missing so far, is – firstly – a *comprehensive* (or at least arbitrarily extensible) set of dedicated simulation preprocessing methods, which are capable of *analyzing, interpreting and processing* all relevant *patient-specific medical data* and which thereby exploit the available *surgical expert knowledge*, in order to set up cardiac surgery simulation scenarios which fit to a mature underlying biomechanical model for simulation-based surgery assistance. Secondly, there is an urgent need for a framework which allows to connect these preprocessing methods into one *combined information processing pipeline* in a single simulation preprocessing step which can be executed *fully-automatically* [MPW⁺16].

A system of this kind, which thus integrates such a comprehensive set of dedicated patient information processing components and which enables automated execution, is imperative for in-OR-usability, since non-engineers and staff in the OR are simply not able to optimally set-up highly complex simulation scenarios. Additionally, also for engineers, researchers and developers, this has the potential to enormously simplify the entire biomechanical modeling workflow and the simulation scenario setup [SSS⁺14, SKS⁺16].

Preferably, the simulation preprocessing step is controlled by some sort of intelligent, cognitive system in order to (possibly fully-automatically, too) set up situation-adapted, patient-specific surgery simulation scenarios [SPW⁺16], see also Section 3.1.5.4. Finally, having set up the biomechanical problem and scenario description during this simulation preprocessing and numerical modeling step, a surgery simulation application (see Section 3.1.5.1) is now able to execute the simulation scenario and to thus produce the virtual surgery outcomes, which are then post-processed by the following postprocessing component (see Section 3.1.5.3).

3.1.5.3 Simulation Postprocessing

In order to make the most out of the results that are obtained from patient-specific biomechanical surgery simulations, and for the surgeon to be able to maximally benefit from the results provided, a simulation postprocessing component is to transform the direct simulation results in an appropriate way. The respective features of interest have to be presented to surgeons such that they can easily get the important properties, interpret their meanings and understand their effects. Moreover, the additional information shall not interfere with established surgical procedures and is preferably integrated seamlessly into clinic workflows.

Dealing with surgery assistance for ring annuloplasty MVR surgery, the most prominent features of surgical interest – according to Carpentier et al. [CAF10] – are reported to be:

1. the post-ring-implantation *MV morphology* throughout the cardiac cycle,
2. the thus resulting *leaflet contact* and size of the coaptation zone at peak systole, and

3. the *stress distribution* across the MV leaflets and annulus at peak systole.

In order for (1) to be presented to the surgeon in surgery assistance, the respective FEM simulations need to be capable of simulating patient-specific annuloplasty ring implantations, which – as presented above in Section 3.1.5.1 – may be handled via appropriate kinematic constraints that account for the annulus shrinking. A subsequent simulation of the cardiac cycle by means of appropriate (blood) pressure constraints then allows to obtain virtual post-ring-implantation MV morphologies, which can be visualized accordingly in order for the surgeon to consider and evaluate them. Votta et al. [VLS⁺13] report on corresponding works.

The second relevant feature of surgical interest, the leaflet contact quantification (2), seems to be directly deducible from the visualization of the simulated MV morphologies (1), however, deteriorated 3D perception, especially during minimally-invasive surgery, often severely complicates the proper recognition and interpretation of leaflet coaptation and its effects on the competence of the MV closure. Choi et al. [CRMK14] propose to not only compute the ring-dependent post-operative size of the coaptation zone throughout the entire cardiac cycle, but to also color-code the leaflet contact area in 4D visualization, such that surgeons can immediately detect non-contact regions in the virtual potential post-ring-implantation results.

With respect to the stress distribution (3), it is important for the operating surgeon to see whether their patient-individual surgical strategy is capable of reducing by means of a specific ring prosthesis the tissue tension and the overall stress on the leaflets and the annulus [CAF10]. If post-ring-implantation stresses for inappropriate rings are above a critical threshold, this may cause rupture of the MV tissue and even lead to a dehiscence of the ring with life threatening consequences. The measure of choice for quantifying stresses has come to be the so-called *von Mises stress*, and most current works support computation and visualization of von Mises stress distributions [MPW⁺16]. Again, Choi et al. prominently raised the state-of-the-art bar by allowing for the patient-specific computation and visualization of pre- and post-ring-implantation stress distributions across the MV leaflets and the annulus at peak systole. They found that in their few manually set-up test cases ring annuloplasty was capable of reducing maximum stresses by a factor of up to 10 [CRMK14].

In order to make complex simulation results, such as the above-mentioned features, understandable for non-simulation-experts and non-engineers, there are, today, several softwares which support and make use of *3D visualization*. Above all, we want to mention ParaView [HAL04] and the Medical Imaging Interaction Toolkit (MITK) [NZS⁺13], both of which are well-established in surgery assistance environments. In particular, the latter one allows to superimpose different features in one view: for instance, the surgeon may look at preoperative 4D ultrasound images of the MV along with a patient-specifically chose annuloplasty ring and with the simulated post-ring-implantation morphologies, on the surfaces of which stress and leaflet coaptation are visualized using color codes. See Figure 3.4 for an example.

Combined with 3D monitors in the OR, this enables surgeons to get a three-dimensional impression of the surgically important properties and allows them to appropriately interpret their respective effects [MVG⁺12]. When it comes to in-OR-integration of such new technologies, there is, however, an ongoing discussion on whether these technologies might interfere or not with established surgical procedures, and possibly even disturb clinical routine [MPW⁺16]. Anyways, it is preferable to design *augmented and virtual realities* in a way such that they pro-

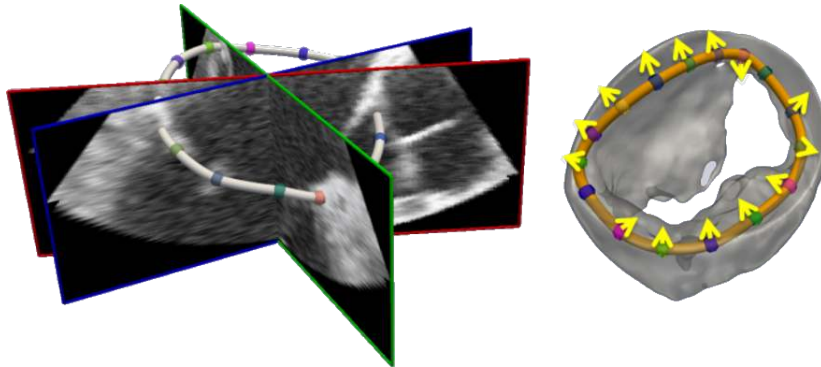


Figure 3.4: Visualization of superimposed 4D ultrasound images of the MV along with a virtually implanted patient-specific annuloplasty ring. [Screenshot of a visualization obtained with the Medical Imaging Interaction Toolkit (MITK).]

vide their additional simulation-based information seamlessly integrated into clinic workflows.

This being said, the *automation* of the respective processes, which allow to provide the above information to surgeons, is imperative for in-OR-usability, and also very convenient for prototyping works, e.g., in the context of biomechanical simulation workflows [SSS⁺14]. We will subsequently focus on this issue in Section 3.1.5.4.

3.1.5.4 Simulation Interface

A big challenge on the way to making FEM-based surgery simulations feasible for in-OR-application is the development of intelligent, automated and efficient *simulation-to-OR interfaces* [MPW⁺16]. In order to improve their in-OR-usability and their overall efficiency, biomechanical simulations must be seamlessly integrated into knowledge-based surgery assistance systems [SEZ⁺15, SPW⁺16].

A simulation interface component is to enable this integration of the surgery simulation application along with its pre- and postprocessing components (and their respective results) into a surgery assistance system and into the OR. This means that the data and information which is obtained pre- and intra-operatively in the OR environment must be identified and interpreted in order to be properly processed through the simulation setup workflow for simulation purposes. Accordingly, after termination of the simulation, the simulation results must be fed back into the OR environment. With respect to the simulation itself, there is a need for an easy and secure access to High-Performance Computing (HPC) resources, as well as their efficient exploitation, such that simulation results can be obtained in adequate times.

However, there is no such automated information processing system to wrap cardiac surgery simulations yet, which – according to the review paper of Ratcliffe et al. [MPW⁺16] – severely hampers the acceptance and in-OR-usability of surgery simulations. The challenging realization of such a system is, among others, certainly due to the fact that – as mentioned in Section 3.1.4 – clinical data often still needs to be digitized and semantified before any intelligent, automated information processing algorithms (be it for simulations or for other purposes) can actually work on the data [FMK⁺16].

Given that this abstract requirement is satisfied, there have been a number of promising approaches in related fields. For instance, Gemmeke et al. [GMP⁺14] have proposed an *information architecture* that is based on linked data and web APIs in order to allow for an automated data consolidation and integration. They exemplarily implemented a pipeline of medical image preprocessing algorithms, however, they appraise their architecture to be transferable to arbitrary chains of individual algorithms, for instance, simulation applications and respective simulation support (pre- and post-processing), too.

Besides that, the company IBM reported in their position paper [BBF⁺15] on research and development towards the *cognitive automation of data science*. They investigate on general information processing by means of methods that are originally settled in the field of Artificial Intelligence, i.e., Machine Learning, etc., and thus, on new possible ways to infer medically and clinically relevant knowledge. Setting up like this patient-specific scenarios for biomechanical simulations, however, is not (yet) intended.

Nevertheless, in a related context, a set of recent works have been presented, which investigate on the impact of the Semantic Web on modeling and simulation [MHC07]. Similarly, Silver et al. [SHM07] proposed a system for basic ontology-driven simulation. However, to the best of the author's knowledge, there has not yet been an application of these concepts to more complex simulation areas such as in the cardiac simulation context. Still, such approaches seem very promising and may afford great opportunities in the above presented context of Surgical Data Science (also see again Section 3.1.4).

Concluding, it must be pointed out that, so far, to the best of the author's knowledge, there have not been any approaches which integrate a biomechanical FEM simulation into a knowledge-based cardiac surgery assistance system, and which thus allow for comprehensively exploiting the available medical knowledge and patient information in order for an *automated, expert-knowledge-based, comprehensive and patient-specific ring annuloplasty simulation*.

For the first time, Schoch et al. [SEZ⁺15] have proposed a concept for such a system. Additionally, a prototypic implementation and works on the respective system components have been presented in dedicated publications [SPW⁺16, SKS⁺16, SES⁺15, SSS⁺14].

Summarizing Sections 3.1.5.1 to 3.1.5.4, it was shown that FEM-based modeling and simulation is a powerful tool for the accurate description of the MV behavior and of pathologic as well as surgical processes in the context of the MV. It can provide insight into the geometry, functionality and stress distributions across the valvular apparatus, which is crucial for a proper understanding of respective diseases and their surgical treatment options. Using such FE models, a variety of simulated surgical repairs has been performed as described in Section 3.1.5.1.

A simulation preprocessing component is to support the setup of comprehensive, patient-specific simulation scenarios by means of analyzing and transforming the available information accordingly for simulation purposes on the one hand. Several approaches which focus on various specific preprocessing issues have been presented; most of them require manual interaction, cf. Section 3.1.5.2. On the other hand, a postprocessing component processes simulation results in a way such that surgeons can maximally benefit from them, e.g., by means of 3D visualization of clinically relevant quantities of interest, cf. Section 3.1.5.3.

Experts believe that these techniques have extensive applications in realm [MPW⁺16]. For

instance, apart from preoperative planning, one could also think of surgical education or optimized device development, etc. However, we are not yet there. To make surgery simulations fit for in-OR-usability, intelligent automated *simulation-to-OR interfaces* have to be developed. These must allow for integrating biomechanical simulations into knowledge-based surgery assistance systems. The most promising approaches were presented in Section 3.1.5.4.

3.2 Open Challenges in the Context of Simulation-based Cardiac Surgery Assistance

As reported in the previous Section, there have been great advances in the context of simulation-enhanced cardiac surgery assistance systems and workflows: A large amount of works have proposed diverse approaches for manually set-up and manually calibrated, partly patient-specific, realistic FEM-based simulations of the MV behavior for cardiac surgery assistance. These works consider both the native case (under natural conditions) and the prosthetic case (after surgical intervention through annuloplasty).

However, the applicability and usefulness of simulation-enhanced MVR surgery support still is controversially discussed. The technology in the presented simulation approaches can not yet fully satisfy the requirement of delivering simulation results that are both highly accurate, realistic and efficiently computed in adequate time, and – in addition to that – intelligently adapted to the current state and course of surgery, therein integrating all available patient information and taking account of surgical expert knowledge as a real human surgeon would do. Thus, we see the following major limitations on the way towards the application of these approaches to surgical planning and general in-OR-usability:

(A) More sophisticated biomechanical models: Current software developments and modeling approaches tackle different areas of interest, such as those reported on in Section 3.1.5. However, besides their respective strengths, they all have (partly severe) disadvantages in items that lay outside these areas. The respective models then are often based on simplifying and generalizing assumptions, which immediately involves that some biomechanical and physiological impact factors get neglected. This claims for the *combination* of the respectively most competent features of these approaches [MPW⁺16], which is particularly important when real conditions in the OR for specific patients become less common than the lesions discussed.

(B) Computational performance / efficiency: Today’s annuloplasty simulations still require computation times of approximately six hours even on massively parallel competent hardware [VLS⁺13, THF⁺13]. There is hence a need for further performance optimization and a more efficient exploitation of the available *High-Performance Computing* (HPC) infrastructure. Also, this limitation may partly be overcome by adopting less time-consuming but still reliable methods, such as the *co-rotated FEM*, which thanks to the linearization of the elasticity equations allows to simulate the MV closure in almost real-time [MVG⁺12], especially when integrated into a powerful computational framework that can take advantage of massively parallel supercomputers, see Sections 3.1.5.1 and 3.1.5.4.

(C) Automated model calibration through data assimilation, adaptivity and sensitivity:

Surgery simulation models can iteratively be improved with respect to selected observation variables or quantities (e.g., the displacement field or stress distributions) through application of Data Assimilation methods [Sul15]. In case of MV simulation, imaging-based observations of an actual MV configuration are therefore incorporated into the model state of the numerical MV model in order to thus allow for a better estimation of respective simulation states or model parameters [PK15]. Generally, through allowing model and simulation *adaptivity* and *sensitivity* with respect to data that is recorded in the OR or that is available in the clinic databases after imaging, one can numerically minimize the error (*calibration*) between real and virtually simulated behavior, i.e., between the observed natural MV and the simulated representation which is based on the respective numerical model, see, e.g., [Ver13]. For instance, chordal effects so far often need to be *manually* calibrated in order to patient-specifically imitate an imaging-recorded leaflet closure behavior [MPW⁺16]. This however may be achieved *automatically* via a proper setup of data assimilation algorithms that are combined with the numerical simulation, the latter of which in turn requires for a competent interface between computer simulation system and OR sensor system, see Sections 3.1.5.2 and 3.1.5.4.

(D) Uncertainty Quantification for surgery simulation: Using methods for *uncertainty quantification* (UQ), one can address questions on the *reliability* and *accuracy* of computer simulation results [SHLM14], which is particularly important in surgery-related simulation applications where surgeons may be influenced in their decision making process by respective simulation results. With an increasing number of publications in the last few years there have been important developments on the theory level. However, for an efficient (possibly real-time) application of UQ in surgery simulation there is a need for dedicated algorithms that exploit mathematical model and simulation properties and additionally make efficient use of optimized HPC methodologies and technologies [LMK10].

(E) Simulation flexibility: It should be kept in mind that *annuloplasty*, which we focus on, is just one option for the surgical treatment of incompetent MVs. Besides that, a number of other techniques has established in surgery, too, some of which have also been successfully simulated on the basis of FEM simulations, e.g., virtual suture, leaflet resection, or subvalvular repairs which may affect the chordae or papillary muscles. In a *flexible, comprehensive* cardiac surgery assistance system, these options would of course need to be integrated, too. This being said, Sturla and colleagues [SOV⁺14] attempted such a prototypic manual investigation of the biomechanical effects (and of their advantages and disadvantages) with respect to different repair techniques. Via manually set-up patient-specific FEM simulations, they found that basically all repair types were able to correct previous mitral regurgitation by restoring leaflet coaptation and relieving tissue stresses. So, the final choice of surgical strategy, which of course will always be up to the surgeons and to their additional intraoperative assessment, will also strongly depend on further pre- or intra-operatively analyzed information and of course on the experience with respective long-term results.

(F) Intelligent, patient-specific, situation-adapted simulation setup: Patient-specific, situation-adapted and surgical expert knowledge-based simulation scenarios are a prerequisite for

a truly beneficial everyday in-OR-application of simulation-based surgery assistance. In order for thus obtaining meaningful simulation results, the simulation scenarios and the underlying biomechanical models must be able to account for and seamlessly integrate all available patient data and consider the best available and best suitable surgical expert knowledge in the respective field [MPW⁺16, BBF⁺15]. This requires *cognitive simulation algorithms* (both with respect to the simulation itself, and its preprocessing, too), which work on the respective data on the one hand, and an *intelligent database management* on the other hand, which facilitates an intelligent usage of the simulation-to-OR interface, e.g., via allowing for semantic queries on this data. This being said, bringing *machine intelligence* into medical technologies generally is a relatively new field of research, and we presented first works in Section 3.1.4, as well as in Sections 3.1.5.2 and 3.1.5.4. However, there still is a great potential (and need) for ongoing and future efforts.

(G) Simulation integration, automation and usability: Currently available automated processes to generate FE models from tomographic images and to thereon-based run simulation scenarios are often incomplete and prohibitively expensive [MPW⁺16]. Also, manually generating any model similar to those described above requires long hours of extensive input by scientists and simulation experts with specific computational and engineering expertise, and there is an additional need for iterative adaptation in a continuous exchange with the respective surgeons. Thus, to foster the *usability* of surgery simulations, these simulations must become fully integrated into cognitive OR assistance environments (*in-OR-integration*) and fully automated within the respective surgery assistance workflows (*automation*). Along with the modeling and simulation component itself, in order for a significant improvement to the usability of surgery simulations, it is hence imperative that the respective pre- and postprocessing components get seamlessly integrated into clinic workflows and into surgical procedures, too. This should preferably be based on a bi-directional exchange via some embedded, cognitive interface: On the one hand, patient information, surgical knowledge and current OR circumstances must find the way into respective simulation scenarios, e.g., via methods of Data Assimilation. On the other hand, the simulation results should directly be fed back into the OR in order to instantaneously provide the surgeon in the right moment with additional information. The idea of such a cognitive simulation-to-OR interface involves that heterogeneous data sources can automatically be queried, analyzed, interpreted and further processed by means of intelligent algorithms, and – again – on the basis of an intelligent data management.

3.3 Concept for Cognition-Guided, Simulation-enhanced Cardiac Surgery Support

As explained in Chapter 2, mitral valve reconstruction (MVR) requires a surgeon to have in-depth experience and a profound medical expert knowledge. During surgery, the surgeon has to keep track of a large number of surgically relevant, patient-specific impact factors in order to select and perform the best suitable surgery strategy. Hence, biomechanical simulations of intelligently set-up MVR annuloplasty surgery scenarios may provide surgeons with valuable additional information, especially during a highly complex surgical operation such as an MVR.

Given the above mentioned state-of-the-art in simulation-based MVR surgery support and

the big open challenges, in the framework of the collaborative research center *Cognition-Guided Surgery* (SFB TRR 125), a particular focus was set onto patient-specificity and cognition-guidance. With respect to simulation-based cardiac surgery assistance, this yields the following requirements list and vision:

An FEM simulation scenario for intraoperative MVR surgery assistance needs to be patient-specific (F), i.e., account for and holistically exploit the available patient information and integrate it into the specific scenario setup.

Further, a useful scenario setup needs to consider current surgical expert knowledge and best surgical practices (F), in order to thereon-based derive a patient-individual surgical strategy, which is moreover adapted to the respective current situation in the OR (F).

Besides these content-related prerequisites, the entire simulation assistance setup is required to be integrated into the traditional MVR surgery process (G), and needs to be provided to the surgeon fully-automatically (G) and in adequate times through efficient algorithms and a high performance computing infrastructure (B).

Note that such a setup is the direct prerequisite for then tackling challenges (C) and (D), too, which can be set on top, once the emerging system prototype is operational. Challenges (A) and (E) can – expressing it a bit exaggeratedly – possibly be seen as diligent but routine pieces of work to be accomplished, since all integral functionalities have already been developed separately (see Section 3.1.5.1), and then have to be stably combined.

Summarizing, the idea is to have a **cognition-guided simulation-based surgery assistance** which – just like a human intelligent assistant – supports MVR surgery by providing the operating surgeon fully-automatically with patient-specific, situation-adapted, expert-knowledge-based, comprehensive MVR simulation scenario results, which are based on a preceding holistic analysis of the available patient information and formalized surgical knowledge.

When operational, such a system can add enormous value for clinical analysis, diagnosis, surgery planning, treatment and training. In the context of my research and development efforts in my work as a PhD student I work towards this idea and aim at obtaining a *prototype* for such a surgery assistance system for the specific application case of *minimally-invasive MVR surgery assistance*.

On the one hand, this system enables the holistic analysis and interpretation of MVR patient information under consideration of expert knowledge and best cardiac surgery practices. On the other hand, it is capable of fully-automatically simulating suitable, patient-individual MVR annuloplasty ring implantation scenarios and their virtual post-surgery outcomes, i.e., the stress and deformation behavior of the MV tissue subject to different ring prostheses. A prototype of such a system is a step towards a *cognition-guided simulation-enhanced cardiac surgery assistance*.

Please note that the needs of cardiac surgeons with respect to intelligent (simulation-based) cardiac surgery assistance also represent a perfectly suitable sample application for the vision and program of the first funding period of the Transregional Collaborative Research Center (TCRC) *Cognition-Guided Surgery* (SFB/TRR 125), which was introduced in Chapter 1.

3.3.1 Requirements for Cognition-guided, Simulation-based Surgery Assistance

In order to actually build such a system/prototype, the entire medical data acquisition and processing chain as in Figure 3.1 has to be taken into account and must become integrated into the system setup process.

In particular, we see the need for the following items and capabilities to be implemented, set-up and employed:

- patient data, which is considered relevant for cardiac surgery has to be represented, modeled and semantically annotated in the framework of a knowledge representation and management system;
- this is to generally allow for querying this data in some data storage system, which has to be set-up and organized in the context of a common clinic software and data infrastructure;
- information processing algorithms have to be developed in order to analyze, interpret and further process all information, i.e., the data along with the knowledge;
- making use of these algorithms, a comprehensive information processing chain/workflow has to be set-up in order to cope with the following aspects:
 - image and segmentation data will be processed;
 - relevant patient-specific parameters need to be filtered, converted and analyzed;
 - reasoning must be performed on this data, e.g., by means of a deductive system for MVR ring selection;
 - heterogeneous data, e.g., image data and data retrieved from parameter data, must be merged and again further processed;
 - suitable 3D models for FEM simulations have to be derived along with appropriate material parameter regions, boundary conditions, etc. (simulation preprocessing);
 - based on the simulation preprocessing, an FEM simulation must be set-up, along with an intelligent simulation scenario management, with an efficient parallelization, with meaningful solvers/preconditioners, etc.;
 - the simulation must be executed on the available hardware compute infrastructure, and the simulation results must be post-processed and fed back into the OR in a way which is beneficial and well-understandable for the surgeon;
- generally, all data (including all kinds of cross-links) and all algorithms (including all of their inputs/outputs, functionalities, parameters) must semantically be described and abstracted with respect to some common ontology, such that a smooth interoperability of all information processing and data enhancement algorithms can be guaranteed;
- a fully-automated workflow can then be set-up as a data-driven pipeline of cognitive applications/algorithms;
- these cognitive applications are to facilitate the evolution of patient-specific medical data for cardiac surgery assistance;

- finally, derived data must be stored (and semantically annotated) again in the central storage system and the respective knowledge base.

This being said, we want to again put particular emphasis on a few very important challenges with respect to the implementation and the employment of this system:

- setting up a *knowledge base* along with the respective semantics-based data and knowledge management, such that heterogeneous patient-individual data can be put in relationship with comprehensive factual knowledge;
- setting up a comprehensive *biomechanical model* and a suitable, stable, efficient and reliable *FEM simulation* to holistically represent the MV behavior and the MVR procedure;
- setting up a comprehensive, knowledge-based *simulation preprocessing pipeline* (of separate intelligent algorithms) in order to process all available data and knowledge for simulation;
- setting up an overall system which combines and connects different softwares, toolkits or libraries via sophisticated interfaces in order for a meaningful *evolution of medical data*;
- setting up an automated, cognitive, data-driven *data processing chain/workflow*.

3.3.2 Context, Goals and Structure of this Work

Concluding, the study of **cognitive assistance systems** is a highly topical field of research, which gains increasing relevance particularly due to the ongoing digitalization of today's society. Especially in the field of medicine, the great potential of large amounts of data and heterogeneous pieces of information is realized at the same time, and there is the strong endeavor to integrate this information in order for the derivation of clinically relevant new insights.

Combining these ideas with biomechanical surgery simulation, and thus setting up cognition-guided, surgical expert knowledge-based, patient-specific surgery simulations, gives reason to expect great new opportunities to support surgeons during complex surgical operations, such as mitral valve reconstructions.

In the framework of my PhD thesis and of the associated research work, I want to thus go a step towards the development of a prototype of such a **cognition-guided, simulation-enhanced cardiac surgery assistance system**.

I investigate on ways how numerical simulations of the MV soft tissue behavior can be set-up and become patient-specific as well as situation-adapted by means of cognitive information processing approaches in the context of a knowledge-based surgery assistance system:

The *FEM-based simulation* is to be maximally parallelized in order to exploit the available high performance computing infrastructure, and optimized with respect to its patient-specific setup. *Patient-specificity*, in turn, shall be achieved through an intelligent, holistic, machine

cognition-based information processing pipeline, which seamlessly collaborates with the simulation preprocessing step and with the mathematical model on which the simulation is built up. The *cognitive information processing pipeline* thereby covers the entire clinical data acquisition and processing spectrum – from imaging via segmentation to parameter recording and the thereon-based clinical reasoning – in order to thus allow for a maximally comprehensive, knowledge-based *evolution of medical data*. This is to thus finally yield the information for comprehensive, intelligent, patient-specific simulation scenarios. Moreover, the cognitive pipeline is to prepare the ground for the full automation of *simulation-enhanced surgery assistance*, and hence fosters the usability of surgery simulations in the OR.

Preferably, through being set-up as a *modular* prototypic research work, the whole system shall be flexibly extensible, and thus allow for future enhancements, such as, e.g., the integration of methods for uncertainty quantification, or a stronger semantic guidance and control through the intensive use of and connection to clinical and numerical ontologies. Like this, the exemplary MVR-focused holistic concept and system prototype is to even become transferable to other fields of surgery.

Given the current state-of-the-art, and knowing about the big, current open challenges in the context of simulation-enhanced cardiac surgery assistance, the intended system and research hence constitutes – to the best of the author’s knowledge – a unique and highly innovative prototypic work which prominently combines the two fields of data science and numeric simulations: For the first time, it integrates an intelligently set-up patient-specific biomechanical surgery simulation and its results into a comprehensive, semantics-based surgery assistance and decision support system.

Proceeding, the **structure of this work** is set-up as follows:

We will focus on the mathematical MV/MVR model and the numerical soft tissue simulation in Chapters 4 and 5. Chapter 6 investigates on the process of setting up respective simulation scenarios in a complex, comprehensive simulation preprocessing pipeline. In Chapter 7, we will explain how by means of cognitive approaches this pipeline can process patient-specific data under consideration of surgical expert knowledge and in the context of a clinical software and data infrastructure. Following, in Chapter 8, we will present numerical and conceptual experiments, which indicate the efficacy of the thus designed cognition-guided, patient-specific, simulation-enhanced cardiac surgery assistance system, and which give a hint at its meaningful application in the *OR of the future*. In Chapter 9, we conclude with a final discussion of our work and with a brief outlook to potential future further developments.

4 Modeling and Simulating the Behavior of the Mitral Valve

As the central element of our simulation-enhanced cardiac surgery assistance system, we will focus in this chapter on the mathematical modeling and numerical simulation of the MV behavior, of the MVR procedure and of its post-operative results. Therefore, in Section 4.1, we introduce basic *physical quantities of elasticity* and important relations for describing and modeling the behavior of soft tissues. Building up on this, in Section 4.2, we *specify* the derived general model equations of contact problems in soft bodies for the particular case of modeling the opening and closing behavior of the MV, both before and after MVR surgery. Finally, in Section 4.3, we look at the *numerical solution of the contact elasticity problem for the MV* by means of the Finite Elements Method (FEM) and different time integration schemes.

4.1 Basics of Elasticity Theory

Since we are interested in the elastic behavior of the mitral valve leaflets, we give a short overview of *continuum mechanics*, and *elasticity theory* in particular. The theory of elasticity describes a soft body's behavior subject to external manipulation. It therefore considers the relationship between the four physical quantities displacement, strain, stress and force. The essential components of elasticity theory are:

- *the kinematics*, which link the displacement of particles in the elastic continuum to the internal deformation measure strain,
- *the constitutive (material) laws*, that link strain to stress, and
- *the equilibrium equations*, which relate the stress to the external loads and forces.

Forces and displacements can be observed as external factors, and hence prescribed by defining corresponding *boundary conditions*.

In the following, we introduce the basic concepts and quantities, define an appropriate notation, and present useful formulations for the elasticity problem, as far as needed for subsequent analysis during the derivation of our actual mitral valve model. Structure- and notation-wise we closely stick to Holzapfel [Hol00].

4.1.1 Kinematics and the Concepts of Deformation, Strain and Stress

4.1.1.1 The Concept of Deformations

A *soft body* \mathcal{B} can be considered as a continuous or at least *piecewise continuous* distribution of matter in 3-dimensional space and in time. In macroscopic studies, both mass and volume are

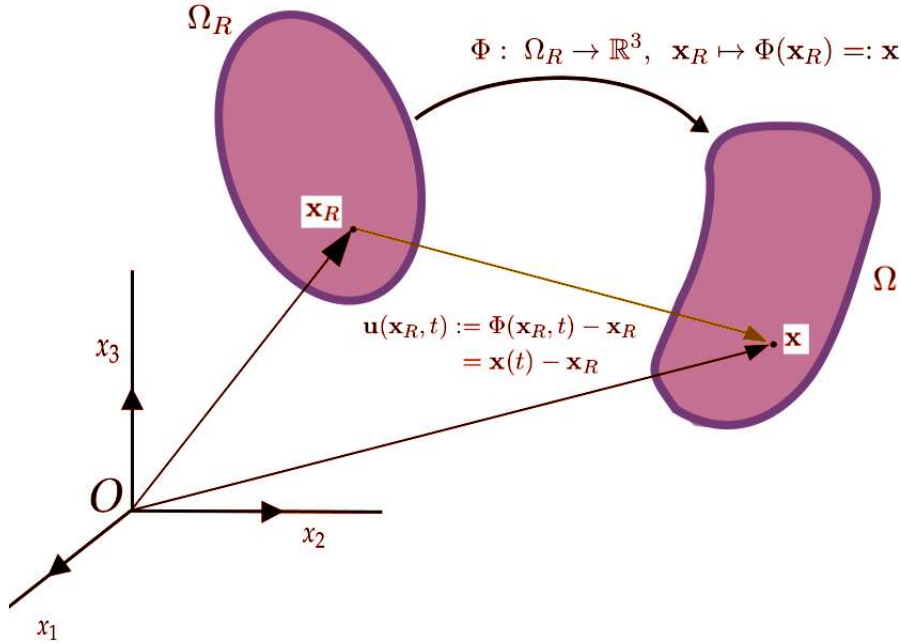


Figure 4.1: Schematic draft of a deformation in continuum mechanics: The *soft body* \mathcal{B} is deformed from its reference configuration Ω_R into a current configuration Ω .

continuous (or at least piecewise continuous) functions of continuum particles. The body moves in space from one point of time to another, occupying different geometrical regions Ω_R, \dots, Ω in this process. These regions, that are bounded subsets $\Omega_i \subset \mathbb{R}^3$, are called configurations of \mathcal{B} at time t . The configuration Ω_R at time $t = 0$ is called the initial (and reference) configuration, while the configuration Ω at time t is called the current configuration.

We introduce a reference frame of right-handed, rectangular coordinate axes at a fixed origin \mathcal{O} with an orthonormal basis. A point \mathbf{x}_R of the reference or initial configuration Ω_R is related to a point \mathbf{x} of the current configuration Ω via the movement of the body \mathcal{B} . For illustration, see Figure 4.1.

Our goal is to describe the deformation of \mathcal{B} with respect to its reference configuration Ω_R . Therefore, we define a mapping

$$\Phi: \Omega_R \rightarrow \mathbb{R}^3, \quad \mathbf{x}_R \mapsto \Phi(\mathbf{x}_R) =: \mathbf{x}, \quad (4.1)$$

which describes the location of an arbitrary point \mathbf{x}_R in the reference configuration (denoted by subscript R) as $\Phi(\mathbf{x}_R) = \mathbf{x}$ in the deformed configuration (without the subscript R). In the following, Φ is always assumed to be uniquely invertible and sufficiently smooth, i.e., it possesses a continuous derivative. It is called a *deformation*, if $\det(\nabla\Phi) > 0$ holds.

Generally, a body undergoing a deformation can change its shape, position and orientation. If the deformation is constant for all $\mathbf{x}_R \in \mathcal{B}$, the deformation consists only of translations and rotations and is called a *rigid-body motion*: We then find $\Phi(\mathbf{x}_R) = \mathbf{Q}\mathbf{x}_R + \mathbf{b}$, where \mathbf{Q} is an orthogonal matrix and \mathbf{b} a translation vector.

This being said, a deformation is often described in terms of the *displacement field* \mathbf{u} of the body \mathcal{B}

$$\mathbf{u}(\mathbf{x}_R, t) := \Phi(\mathbf{x}_R, t) - \mathbf{x}_R = \mathbf{x}(t) - \mathbf{x}_R, \quad (4.2)$$

or in other words

$$\Phi = \text{id} + \mathbf{u}. \quad (4.3)$$

The gradient of the deformation $\nabla_R \Phi$ is called *deformation gradient (tensor)*, and its matrix notation reads as

$$\mathbf{F}(\mathbf{x}_R, t) := \nabla_R \Phi(\mathbf{x}_R, t) = \begin{bmatrix} \frac{\partial \Phi_1}{\partial x_1} & \frac{\partial \Phi_1}{\partial x_2} & \frac{\partial \Phi_1}{\partial x_3} \\ \frac{\partial \Phi_2}{\partial x_1} & \frac{\partial \Phi_2}{\partial x_2} & \frac{\partial \Phi_2}{\partial x_3} \\ \frac{\partial \Phi_3}{\partial x_1} & \frac{\partial \Phi_3}{\partial x_2} & \frac{\partial \Phi_3}{\partial x_3} \end{bmatrix} = \text{grad}_R \Phi(\mathbf{x}_R, t) = \text{grad}_R \mathbf{u}(\mathbf{x}_R, t) + \mathbf{I}. \quad (4.4)$$

The determinant $J := \det(\nabla \Phi)$ of the deformation gradient is the local ratio of current volume to reference volume of a material volume element. By virtue of the impenetrability of matter and the non-singularity of \mathbf{F} , we find $J \equiv \det(\nabla \Phi) > 0$, and in particular $J = 1$ for incompressible materials in the context of soft tissue modeling.

Besides characterizing the deformation of volume elements, the deformation gradient of course also allows to describe the deformation of infinitesimal line and area elements during the body motion. For more details, we refer to [Hol00].

4.1.1.2 The Concept of Strain

In order to define constitutive laws for describing the behavior of soft materials and soft tissues, it is necessary to determine a 3D equivalent of stretch, the so-called *strain*, inside a body. The strain can be explained as a measure for the change in length of infinitesimal line elements. Unlike displacements, which are measurable quantities, strain is not necessarily a physically measurable quantity, but rather a theoretical concept. Consequently, many different strain measures exist, the most important of which (in the context of soft tissue modeling) we present in the following.

Firstly, one might think of using the above introduced deformation gradient tensor \mathbf{F} as a strain measure or strain tensor. However, it is unsuitable as strain tensor as can be seen from the *polar decomposition theorem* (see, e.g., in [Hol00], and a proof in [Ogd97]), which states that any non-singular second order tensor \mathbf{A} can be decomposed and represented by a unique positive definite symmetric second order tensor \mathbf{U} and an orthogonal second-order tensor \mathbf{R} such that $\mathbf{A} = \mathbf{R}\mathbf{U}$ holds.

For the non-singular deformation gradient tensor \mathbf{F} this means that it can be decomposed into a pure rotation matrix \mathbf{R} and a pure stretch matrix \mathbf{U} . Consequently, \mathbf{F} changes under pure rigid body motions. However, obviously, a reasonable strain measure should be invariant under rigid body motions. Therefore, \mathbf{F} cannot directly be used as a strain tensor.

To recover rotational invariance, one option is to perform a polar decomposition and to use the remaining stretch matrix as a strain tensor. In classical solid mechanics, a quadratic strain measure is used, as it yields nice analytic analysis of the ensuing equations. Hence, inserting

the deformation gradient tensor \mathbf{F} into the so-called *Cauchy-Green strain tensor*

$$\mathbf{C} = \mathbf{F}^T \mathbf{F} \quad (4.5)$$

we obtain a rotationally invariant tensor (due to orthogonality of \mathbf{R}):

$$\mathbf{C} = \mathbf{F}^T \mathbf{F} = (\mathbf{R}\mathbf{U})^T \mathbf{R}\mathbf{U} = \mathbf{U}^T \underbrace{\mathbf{R}^T \mathbf{R}}_{=\mathbf{I}} \mathbf{U} = \mathbf{U}^T \mathbf{U}. \quad (4.6)$$

Another important and well-established strain measure is the related *Green-Lagrange strain tensor*, which is defined as

$$\begin{aligned} \mathbf{E} &= \frac{1}{2}(\mathbf{C} - \mathbf{I}) = \frac{1}{2}(\mathbf{F}^T \mathbf{F} - \mathbf{I}) = \frac{1}{2} \left((\nabla \mathbf{u} + \mathbf{I})^T (\nabla \mathbf{u} + \mathbf{I}) - \mathbf{I} \right) \\ &= \frac{1}{2} \left(\nabla \mathbf{u}^T + \nabla \mathbf{u} + \nabla \mathbf{u}^T \nabla \mathbf{u} \right). \end{aligned} \quad (4.7)$$

It is also symmetric and rotationally invariant. Note that both tensors are nonlinear, and that this is needed for rotational invariance.

For later analysis in the context of linear elasticity theory, we list the *infinitesimal Cauchy strain tensor*

$$\boldsymbol{\varepsilon} = \frac{1}{2} \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right), \quad (4.8)$$

which is obtained from the finite Green-Lagrange strain tensor \mathbf{E} when neglecting the quadratic term, and which hence represents the linearization of the Green-Lagrange strain tensor.

We remark at this point, that the above mentioned polar decomposition will be referred to later again in the context of the corotational elasticity model.

4.1.1.3 The Concept of Stress

Motion and deformation give rise to interactions between adjacent material parts in the interior of the body \mathcal{B} . One consequence of these interactions is stress, which has the physical dimension force per unit of area. The notion of stress, which is responsible for the deformation of materials, is crucial in continuum mechanics. We consider a deformable body during a finite motion, and discuss the properties of traction vectors and stress tensors in different descriptions.

Cauchy postulated the existence of a *surface traction vector* $\mathbf{t} = \mathbf{t}(\mathbf{x}, \partial\Omega)$, and that it has the same value for all boundaries with the same normal direction \mathbf{n} , see [Hol00]. In other words, \mathbf{t} only depends on the surface normal \mathbf{n} and we can write:

$$\mathbf{t}(\mathbf{x}, \partial\Omega) = \mathbf{t}(\mathbf{x}, \mathbf{n}). \quad (4.9)$$

From this postulate, one can establish *Cauchy's fundamental stress theorem*, which states that the stress vector $\mathbf{t}(\mathbf{x}, \mathbf{n})$ depends linearly on \mathbf{n} , provided that it is continuous in \mathbf{x} . This means that there exists a second order tensor field $\boldsymbol{\sigma}$ independent of \mathbf{n} , such that

$$\mathbf{t}(\mathbf{x}, \mathbf{n}) = \boldsymbol{\sigma}(\mathbf{x})\mathbf{n} \quad (4.10)$$

for all \mathbf{x} in \mathcal{B} . The tensor $\boldsymbol{\sigma}$ is called the *Cauchy stress tensor*, or *true stress tensor*. A proof

for this theorem can be found, e.g., in [Ogd97] or [BW08].

As a direct consequence of this theorem, we can express a surface force

$$d\mathbf{f} = \mathbf{t}(\mathbf{x}, \mathbf{n})dA = \boldsymbol{\sigma}(\mathbf{x})\mathbf{n}dA \quad (4.11)$$

on an infinitesimal surface element (the so-called surface traction) in the current configuration using the Cauchy stress tensor and the surface normal.

We remark at this point that the below listed conservation law for rotational momentum, see (eq. 4.21), implies the symmetry of the Cauchy stress tensor. For more details, we refer to [Hol00]. Also, we point out that the Cauchy stress tensor $\boldsymbol{\sigma}$ is the stress measure from which the so-called *von Mises stress* (or *yield stress*) is derived. The von Mises stress σ_{vM} is a scalar value, which is used to predict the yielding of a material under loading conditions, and the respective von Mises yield criterion suggests that the yielding of a material begins when a critical von Mises stress value is reached. We will get back to the von Mises stress below in the context of the simulation post-processing, see Section 6.3, where the von Mises stress is visualized on the surface of the MV leaflets in order to give surgeons an idea of possibly negative post-operative effects of surgical intervention.

Further, it is to be noted that the above equations have so far been formulated in the current deformed configuration. However, in a typical scenario, the deformed geometry of the body \mathcal{B} is actually the solution that a problem shall be solved for. It is thus impossible to compute quantities with respect to the current configuration or to integrate over it.

To overcome this problem all forces must be related to the reference configuration, and the respective equations then need to be solved using the so-called material description (i.e., with respect to the reference configuration). We use Nanson's formula

$$d\mathbf{A} = J\mathbf{F}^{-T}d\mathbf{A}_R, \quad (4.12)$$

which describes the deformation of the arbitrary surface element $d\mathbf{A}_R$ through the deformation gradient tensor \mathbf{F} . Like this, we facilitate the problem formulation of (eq. 4.11) with respect to the reference configuration. We may thus relate the surface force $d\mathbf{f}$ to the undeformed surface element $d\mathbf{A}_R$ by means of

$$d\mathbf{f} = \boldsymbol{\sigma}(\mathbf{x})\mathbf{n}dA = \underbrace{\boldsymbol{\sigma}J\mathbf{F}^{-T}}_{=: \mathbf{P}}\mathbf{n}_RdA_R = \mathbf{P}\mathbf{n}_RdA_R. \quad (4.13)$$

Here, we first described the infinitesimal surface element $d\mathbf{A} = \mathbf{n}dA$ in the current configuration in terms of its surface normal \mathbf{n} , and analogously wrote $d\mathbf{A}_R = \mathbf{n}_RdA_R$ with the surface normal \mathbf{n}_R in the reference configuration. Second, we introduced the *first Piola-Kirchhoff stress tensor*

$$\mathbf{P} = J\boldsymbol{\sigma}\mathbf{F}^{-T} \quad (4.14)$$

which relates surface forces in the current configuration to surface elements in the reference configuration. The passage from $\boldsymbol{\sigma}$ to \mathbf{P} is often referred to as the *Piola transformation*. Following this notation, a so-called *push-forward operation* transforms a vector-valued or tensor-valued quantity that is given with respect to the reference configuration into its description with respect to the current configuration, and a *pull-back* is an inverse operation, which transforms

a vector-valued or tensor-valued quantity based on the current configuration to the reference configuration.

We point out that, in contrast to the Cauchy stress tensor $\boldsymbol{\sigma}$, the first Piola-Kirchhoff tensor \mathbf{P} is not symmetric (as the deformation gradient \mathbf{F} is generally not symmetric either). For this reason, we here introduce one more stress measure (among many others proposed in the literature), the symmetric *second Piola-Kirchhoff stress tensor*

$$\mathbf{S} = J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T} = \mathbf{F}^{-1}\mathbf{P} = \mathbf{S}^T, \quad (4.15)$$

which is very important in the context of soft tissue simulations, compare section 4.1.3. We remark that for small deformations, the differences between the three stress tensors $\boldsymbol{\sigma}$, \mathbf{P} and \mathbf{S} are negligible, cf. [Bra07].

4.1.2 Equilibrium Equations and the Boundary Value Problem of Elasticity

4.1.2.1 Equilibrium Equations

In all branches of continuum mechanics, the behavior of objects is governed by four fundamental conservation laws: The conservation of mass, the balance of linear and angular momentum, and the conservation of energy. They are applicable to any particular material and must be satisfied for all times.

Every continuum body \mathcal{B} possesses mass, denoted by m , which is a fundamental physical property commonly defined to be a measure of the amount of a material contained in the body. We assume that the mass is continuously (or at least piecewise continuously) distributed over an arbitrary region $\Omega \subset \mathbb{R}^3$ at time t . Even if the body \mathcal{B} occupies different geometrical regions Ω_R, \dots, Ω over time, it is a closed system and its mass m does not change. We denote the density of \mathcal{B} with $\rho_R(\mathbf{x}_R)$ in the reference configuration, and the density in the current configuration with $\rho(\mathbf{x})$. Using this notation, the *conservation of mass* writes as

$$m = \int_{\Omega_R} \rho_R(\mathbf{x}_R) dV_R = \int_{\Omega} \rho(\mathbf{x}) dV = \text{const.} > 0. \quad (4.16)$$

In order to describe the balance of linear and angular momentum, we consider a closed system with a given motion \mathbf{x} (and the respective displacement \mathbf{u}), mass density $\rho = \rho(\mathbf{x}, t)$ and a velocity field $\mathbf{v} = \mathbf{v}(\mathbf{x}, t)$. The linear momentum in the current configuration then reads as

$$\int_{\Omega} \rho \mathbf{v} dV = \int_{\Omega} \rho \dot{\mathbf{x}} dV. \quad (4.17)$$

Be aware that we denote the time derivative $\frac{\partial}{\partial t} \mathbf{x}$ by $\dot{\mathbf{x}}$, and analogously $\frac{\partial^2}{\partial t^2} \mathbf{x} = \ddot{\mathbf{x}}$. Moreover, we define $\dot{\mathbf{x}}_R := \dot{\mathbf{x}}|_{x_R, t=0}$ and $\ddot{\mathbf{x}}_R := \ddot{\mathbf{x}}|_{x_R, t=0}$.

The linear momentum is changed, when \mathcal{B} is subjected to external forces. In this context, so called volumetric body forces that act on the body's volume (e.g. gravity) are distinguished from surface or contact forces that act on the body's surface (e.g. blood pressure on tissue surfaces). Using the gravitational constant \mathbf{g} , the gravitational body force (in its integral form) reads as

$$\int_{\Omega} \rho \mathbf{g} dV. \quad (4.18)$$

Analogously, the integral description for the surface or contact force is defined by

$$\int_{\partial\Omega} \mathbf{t}(\mathbf{x}, \partial\Omega) dA, \quad (4.19)$$

where again $\mathbf{t}(\mathbf{x}, \partial\Omega)$ denotes the surface traction vector (or surface force density) as in (eq. 4.9).

Summarizing, the *balance of linear momentum* writes as

$$\int_{\partial\Omega} \mathbf{t}(\mathbf{x}, \partial\Omega) dA + \int_{\Omega} \rho \mathbf{g} dV \equiv \frac{d}{dt} \int_{\Omega} \rho \mathbf{v} dV = \int_{\Omega} \rho \dot{\mathbf{v}} dV = \int_{\Omega} \rho \ddot{\mathbf{x}} dV. \quad (4.20)$$

Note that for a body to be in complete equilibrium it is not sufficient that all internal and external forces acting on the body are in balance. Even if all external forces cancel out (i.e., in case of static equilibrium), the body can still be subjected to a rotational motion, if these forces act on different points of the body. The rotational equilibrium is ensured by the *balance of rotational momentum*. By defining the position vector $\mathbf{r}(\mathbf{x}) = \mathbf{x} - \mathbf{x}_R$ relative to a fixed point \mathbf{x}_R it can be formulated as

$$\int_{\partial\Omega} \mathbf{r} \times \mathbf{t}(\mathbf{x}, \partial\Omega) dA + \int_{\Omega} \mathbf{r} \times \rho \mathbf{g} dV \equiv \frac{d}{dt} \int_{\Omega} \mathbf{r} \times \rho \mathbf{v} dV = \int_{\Omega} \mathbf{r} \times \rho \dot{\mathbf{v}} dV = \int_{\Omega} \mathbf{r} \times \rho \ddot{\mathbf{x}} dV. \quad (4.21)$$

The combination of the equilibrium equations (eq. 4.20) and (eq. 4.21) is described in the so-called *axiom of static equilibrium*. We remark that the principle of linear momentum conservation will be used to derive the basic boundary value problem of elasticity (see Section 4.1.2.2), and the principle of rotational momentum conservation yields the symmetry of the above introduced Cauchy stress tensor, compare (eq. 4.10).

In the context of thermodynamics the *conservation of mass* and the *balance of momentum* are usually supplemented by the *conservation of energy*. However, in the realm of soft tissue simulation the thermal effects of the body motion are extremely small and therefore most commonly neglected, cf. [Hol00]. In this work, we hence assume that body motion can be fully described using the three above-described balance principles.

4.1.2.2 The Boundary Value Problem of Elasticity

Before considering the constitutive (material) laws, only accounting for the equations of kinematics and the conservation laws, we can combine the conservation laws into a single partial differential equation (PDE). Along with appropriate boundary conditions, this PDE then yields a boundary value problem (BVP).

Inserting the Cauchy stress tensor $\boldsymbol{\sigma}$ as in (eq. 4.10) into the balance of linear momentum (eq. 4.20), we obtain

$$\int_{\partial\Omega} \boldsymbol{\sigma}(\mathbf{x}) \mathbf{n} dA + \int_{\Omega} \rho \mathbf{g} dV = \int_{\Omega} \rho \ddot{\mathbf{x}} dV. \quad (4.22)$$

Applying Gauss' divergence theorem (see, e.g., [Bra07]) to the surface term, we derive

$$\int_{\Omega} \operatorname{div} \boldsymbol{\sigma}(\mathbf{x}) dV + \int_{\Omega} \rho \mathbf{g} dV = \int_{\Omega} \rho \ddot{\mathbf{x}} dV, \quad (4.23)$$

which is also known as *Cauchy's first equation of motion*. This relation, in its global form as

in (eq. 4.23) with respect to the whole body \mathcal{B} , is supposed to hold for any arbitrary volume dV in the body \mathcal{B} . Hence, we can deduce the differential local form

$$\operatorname{div}\boldsymbol{\sigma}(\mathbf{x}) + \rho\mathbf{g} = \rho\ddot{\mathbf{x}}, \quad (4.24)$$

for each point $\mathbf{x} \in \Omega$ and for all times t .

Generally, relation (eq. 4.24) is nonlinear in the displacement field \mathbf{u} of the deformation. The nonlinearities are implicitly present due to geometric sources (i.e., the kinematics of motion of the body) and material sources (i.e., the material itself, as the Cauchy stress $\boldsymbol{\sigma}$ may, in general, depend on \mathbf{u} , as will be presented below).

If the acceleration is assumed to be zero for all $\mathbf{x} \in \Omega$, which explicitly does not exclude a constant velocity field, we find that (eq. 4.24) simplifies as

$$\operatorname{div}\boldsymbol{\sigma}(\mathbf{x}) + \rho\mathbf{g} = \mathbf{0}, \quad (4.25)$$

which is referred to as *Cauchy's equation of equilibrium* in electrostatics.

As stated above, the PDE (eq. 4.24) cannot be solved when formulated in terms of the current (unknown) configuration. Therefore, we use the mass conservation (eq. 4.16) and Nanson's formula (eq. 4.12) to express (eq. 4.22) with respect to the reference configuration:

$$\int_{\partial\Omega_R} \boldsymbol{\sigma} J \mathbf{F}^{-T} \mathbf{n}_R dA_R + \int_{\Omega_R} \rho_R \mathbf{g} dV_R = \int_{\Omega_R} \rho_R \ddot{\mathbf{x}}_R dV_R. \quad (4.26)$$

Inserting the first Piola-Kirchhoff stress tensor (eq. 4.14) and again applying Gauss' divergence theorem analogously yields Cauchy's first equation of motion with respect to the reference configuration

$$\int_{\Omega_R} \operatorname{div}_R \mathbf{P} dV_R + \int_{\Omega_R} \rho_R \mathbf{g} dV_R = \int_{\Omega_R} \rho_R \ddot{\mathbf{x}}_R dV_R, \quad (4.27)$$

and the corresponding local formulation of the boundary value problem

$$\operatorname{div}_R \mathbf{P} + \rho_R \mathbf{g} = \rho_R \ddot{\mathbf{x}}_R \quad \forall \mathbf{x}_R \in \Omega_R. \quad (4.28)$$

In order to set up a physically meaningful problem, additional *boundary conditions (BCs)* have to be specified on the elastic body \mathcal{B} , to supplement the above balance equation. See Figure 4.2 for illustration of the ensuing definitions. The parts of the surface $\Gamma_D \subseteq \partial\Omega$ where the displacement \mathbf{u} (or the position \mathbf{x}) of the body is known are called Dirichlet boundary conditions. In contrast, surface tractions \mathbf{t} are imposed on the Neumann boundary $\Gamma_N := \partial\Omega \setminus \Gamma_D$. In order for the problem to be well-posed, either Dirichlet or Neumann boundary conditions have to be prescribed on the whole boundary ($\partial\Omega = \Gamma_D \cup \Gamma_N$). Furthermore, Γ_D and Γ_N are not allowed to overlap, i.e., $\Gamma_D \cap \Gamma_N = \emptyset$. If in a real world problem the latter condition is broken on some part of the body's boundary $\partial\Omega$, we let the Dirichlet BC be stronger by definition and neglect the impact of the Neumann BC. On Γ_N we again distinguish between an effective boundary (where $\mathbf{t}_N \neq \mathbf{0}$) and a zero boundary (where $\mathbf{t}_N = \mathbf{0}$), and impose $\mathbf{t}_N = \mathbf{0}$ on $\Gamma_N \setminus \Gamma_N^{\neq 0}$ by definition.

Thereon-based, we specify the following function spaces with respect to the above boundary

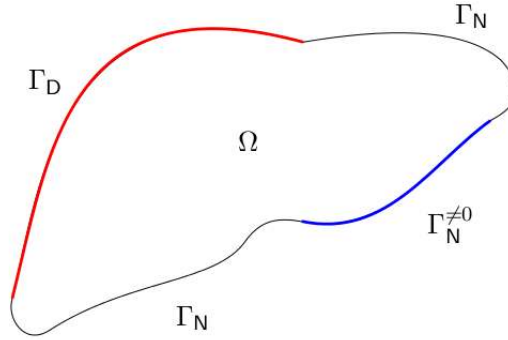


Figure 4.2: Schematic draft of a body Ω along with the different boundary parts $\Gamma_D, \Gamma_N, \Gamma_N^{\neq 0}$. The red line represents the Dirichlet boundary part. The Neumann boundary part constitutes the non-red-lined remaining boundary parts, whereby the blue-lined part specifically represents the effective Neumann boundary part.

conditions:

$$V_D(\Gamma_D, \mathbf{u}_D) := \{\mathbf{x} | \mathbf{u}(\mathbf{x}_R) = \mathbf{u}_D \quad \forall \mathbf{x}_R \in \Gamma_D\} \quad (4.29)$$

$$V_N(\Gamma_N, \mathbf{t}_N) := \{\mathbf{x} | (\boldsymbol{\sigma}(\mathbf{x}(\mathbf{x}_R))\mathbf{n}) = \mathbf{t}_N \quad \forall \mathbf{x}_R \in \Gamma_N\} \quad (4.30)$$

We can then state the *boundary value problem (BVP) of nonlinear elasticity*:

$$\text{Find } \mathbf{x} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N) \quad (4.31)$$

such that (eq. 4.28) holds.

In some literature, the BVP (eq. 4.31) is formally extended by *initial value conditions*

$$\mathbf{x}(t=0) = \mathbf{x}_R \quad \forall \mathbf{x}_R \in \Omega_R \quad \text{and} \quad \dot{\mathbf{x}}(t=0) = \dot{\mathbf{x}}_R \quad \forall \mathbf{x}_R \in \Omega_R \quad (4.32)$$

and the system arising from (eq. 4.31) along with (eq. 4.32) is known as the *Lamé-Navier initial boundary value problem (IBVP) of elasticity*.

Note that so far no constitutive (or material) laws enter *a priori* the above equations. Therefore, they are valid for all problem classes in biomechanics, including elasticity, plasticity, friction or non-conservative loading.

4.1.3 Constitutive (Material) Laws for Biological Soft Tissue

In order to compute for prescribed external forces a body's deformation and stress behavior, we want to solve the above BVP. However, it cannot be solved without so-called *constitutive equations* that relate the position \mathbf{x} (or the displacement \mathbf{u}) to the stress tensor values, and

therein account for the specific material properties. When dealing with (elastic) soft bodies, this (stress-strain) relationship is known as *response function*.

In the following paragraphs, we will therefore give a brief overview on different characteristics of this relation and introduce the notation needed thereafter. Firstly, we will define the concept of the *internal elastic energy* that is stored in the elastic body \mathcal{B} during the deformation. This will lead us to the so-called *work conjugacy* relationship, which links some of the above introduced strain and stress measures and forms special (work conjugate) pairs. Also, the concept of the internal elastic energy will allow for the definition of *hyperelasticity*, which establishes the most important class of nonlinear material models for biomechanics. We will briefly depict basic techniques for modeling *viscoelastic behavior*, which accounts for phenomena like damping, and finally explain under which assumptions the nonlinear elasticity problem reduces to a *linear problem*, which is especially important from the real-time computation point of view.

4.1.3.1 Mechanical Energy in Elastic Soft Bodies

If the material response is purely elastic and no energy is dissipated as heat, the balance of mechanical energy can be directly derived from Cauchy's equation of motion. Therefore, firstly, we multiply the equation of motion (eq. 4.24) with the velocity \mathbf{v}

$$\operatorname{div} \boldsymbol{\sigma} \cdot \mathbf{v} + \rho \mathbf{g} \cdot \mathbf{v} = \rho \ddot{\mathbf{x}} \cdot \mathbf{v}, \quad (4.33)$$

and use the product rule to transform the equation into

$$\operatorname{div}(\boldsymbol{\sigma} \mathbf{v}) - \boldsymbol{\sigma} : \nabla \mathbf{v} + \rho \mathbf{g} \cdot \mathbf{v} = \rho \ddot{\mathbf{x}} \cdot \mathbf{v}. \quad (4.34)$$

Here, we introduced the so-called *Frobenius (inner) product*

$$\mathbf{A} : \mathbf{B} := \sum_{i,j} A_{ij} B_{ij} = \operatorname{trace}(\mathbf{A}^T \mathbf{B}). \quad (4.35)$$

We remark that, generally, an operation on one or more tensors, that arises from the natural pairing of a finite-dimensional vector space and its dual, is denoted as a *(double) tensor contraction*, and can be seen as a generalization of the trace operator.

Secondly, we integrate over the volume of the current configuration

$$\int_{\Omega} \operatorname{div}(\boldsymbol{\sigma} \mathbf{v}) dV - \int_{\Omega} \boldsymbol{\sigma} : \nabla \mathbf{v} dV + \int_{\Omega} \rho \mathbf{g} \cdot \mathbf{v} dV = \int_{\Omega} \rho \ddot{\mathbf{x}} \cdot \mathbf{v} dV, \quad (4.36)$$

Here, we transform the inertia term using the product differentiation rule

$$\rho \ddot{\mathbf{x}} \cdot \mathbf{v} = \rho \dot{\mathbf{v}} \cdot \mathbf{v} = \rho \frac{1}{2} \dot{\mathbf{v}} \cdot \dot{\mathbf{v}} = \rho \frac{d}{dt} \frac{1}{2} \mathbf{v}^2, \quad (4.37)$$

and exploit the symmetry of $\boldsymbol{\sigma}$, which yields

$$\boldsymbol{\sigma} : \nabla \mathbf{v} = \boldsymbol{\sigma} : \underbrace{\left[\frac{1}{2} (\nabla \mathbf{v} + (\nabla \mathbf{v})^T) \right]}_{=:\dot{\boldsymbol{\varepsilon}}} = \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}}, \quad (4.38)$$

with the *rate of deformation tensor* $\dot{\boldsymbol{\varepsilon}}$, which is not to be mixed up with the time derivative of the infinitesimal Cauchy strain tensor $\frac{\partial}{\partial t} \boldsymbol{\varepsilon} = \dot{\boldsymbol{\varepsilon}}$, as will be referred to below in (eq. 4.58). Finally, we have

$$\int_{\Omega} \operatorname{div}(\boldsymbol{\sigma} \mathbf{v}) dV - \int_{\Omega} \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}} dV + \int_{\Omega} \rho \mathbf{g} \cdot \mathbf{v} dV = \frac{d}{dt} \int_{\Omega} \frac{1}{2} \rho \mathbf{v}^2 dV. \quad (4.39)$$

Applying the divergence theorem and using the definition of the surface traction (eq. 4.11) yields the *balance equation for mechanical energy* with respect to the current configuration:

$$\frac{d}{dt} \underbrace{\int_{\Omega} \frac{1}{2} \rho \mathbf{v}^2 dV}_{=\mathcal{W}_{\text{kin}}} + \underbrace{\int_{\Omega} \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}} dV}_{=\mathcal{P}_{\text{int}}} = \underbrace{\int_{\partial\Omega} \mathbf{t} \cdot \mathbf{v} dA}_{=\mathcal{P}_{\text{ext}}} + \int_{\Omega} \rho \mathbf{g} \cdot \mathbf{v} dV. \quad (4.40)$$

Here, \mathcal{P}_{ext} denotes the external mechanical power or rate of external mechanical energy (i.e., the power input on the region Ω), \mathcal{P}_{int} stands for the stress power or rate of internal mechanical energy, and \mathcal{W}_{kin} represents the kinetic energy (i.e., the generalization of Newtonian mechanics to continuum mechanics). If \mathcal{W}_{kin} is zero, and if there are hence no inertia forces, then the dynamic BVP reduces to a *static nonlinear problem*.

For completeness, we also list the *balance equation for mechanical energy* with respect to the reference configuration:

$$\frac{d}{dt} \int_{\Omega_R} \frac{1}{2} \rho_R \mathbf{v}^2 dV_R + \int_{\Omega_R} \mathbf{P} : \dot{\mathbf{F}} dV_R = \int_{\partial\Omega_R} \mathbf{T} \cdot \mathbf{v} dA_R + \int_{\Omega_R} \rho_R \mathbf{g} \cdot \mathbf{v} dV_R, \quad (4.41)$$

with the first Piola-Kirchhoff traction vector \mathbf{T} defined by $\mathbf{T} dA_R = \mathbf{t} dA$. We omit the derivation and refer to Holzapfel [Hol00] for more details.

Also without proof and derivation, we reformulate the above and summarize common representations of the *rate of internal (mechanical) energy* or *stress power* per unit reference volume

$$\mathcal{W}_{\text{int}} = J \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}} = \mathbf{P} : \dot{\mathbf{F}} = \mathbf{S} : \dot{\mathbf{E}} \quad (4.42)$$

of a material, therein listing the most important couples of so-called *work-conjugate pairs* of stress tensors and associated strain rate tensors. Again, we refer to [Hol00] for more details.

4.1.3.2 Hyperelasticity

Usually, biological soft tissue is modeled using a phenomenological approach. As such, mathematical models that describe the stress-strain relationship are fitted to experimental data, which is based on in vitro or in vivo measurements. For an extensive overview and comparison of various material models, we refer to Holzapfel [Hol00] or the work of Truesdell and Noll [TN04]. For details on the most common models, see, e.g., the works by Holzapfel and

Odgen [HO16], Odgen [Ogd97], Hughes [MH83], Mooney [Moo40], and Ateshian [Ate16].

The most important class of nonlinear material models for soft tissues is established by the so-called hyperelastic material model, which is based on the above-mentioned concept of the internal elastic energy. Even though we only discuss isotropic hyperelastic materials at this point, the approach can be extended to include anisotropic material behavior [Hol00]. The extension of the model to viscoelastic material response is depicted in the subsequent section.

If a material's stress field in the deformed configuration only depends on the state of deformation and not on the deformation history, a so-called *response function* $\mathcal{R}(\mathbf{F}(\mathbf{x}_R, t), \mathbf{x}_R)$ can be defined, and relate the deformation gradient tensor \mathbf{F} to the Cauchy stress field $\boldsymbol{\sigma}$:

$$\boldsymbol{\sigma}(\mathbf{x}, t) := \mathcal{R}(\mathbf{F}(\mathbf{x}_R, t), \mathbf{x}_R). \quad (4.43)$$

For example, the above-mentioned *Piola transformation*, see (eq. 4.14), relates the deformation to the first Piola-Kirchhoff stress tensor:

$$\mathbf{P} = J\boldsymbol{\sigma}\mathbf{F}^{-T} = J\mathcal{R}(\mathbf{F})\mathbf{F}^{-T} =: \mathcal{H}(\mathbf{F}). \quad (4.44)$$

Materials for which a relation as in (eq. 4.43) exists, are called *Cauchy-elastic*.

In order for finding suitable material-specific response functions for biological soft tissue, the standard approach goes via the description of the *internal energy* of a material. Therefore, the existence of a so-called *elastic potential* or *strain energy function* Ψ is postulated. It is a scalar-valued function which is defined per unit reference volume, and it describes the internal energy of a material. In literature, the strain energy function is also often referred to as *free-energy function* or *stored energy function*.

Materials for which Ψ exists and only depends on the deformation (i.e., $\Psi = \Psi(\mathbf{F})$) and not on the deformation history are called (*pure*) *hyperelastic materials*. In contrast to Cauchy-elasticity, hyperelasticity hence additionally claims that not only the stress field but also the internal energy of the deformed configuration is independent of the deformation history or the deformation path [Cia88].

For work-conjugate stress-strain tensor pairs, elastic response functions can be derived on the basis of the elastic potential Ψ . At this point, we only outline this procedure and list the most important results for further usage in our work. For more details we refer to Holzapfel [Hol00].

Usually, the starting point is the time derivative of the elastic potential

$$\dot{\Psi} = \mathcal{W}_{\text{int}} = \mathbf{P} : \dot{\mathbf{F}}, \quad (4.45)$$

which is the internal work, and which can be expressed through work-conjugate pairs as listed in (eq. 4.42).

Obviously, material models should be constructed in a way such that the BVP (eq. 4.31) has an (ideally unique) solution that corresponds to physical observation. Motivated by the physical behavior of soft materials, we find that, when subjected to external forces, real materials deform in a way that minimizes the strain energy function Ψ , compare, e.g., [HN03] or [MH83].

The mathematical conditions under which such a Ψ which can be minimized with respect to \mathbf{F} exists, are twofold: The first obvious and important requirement is that the strain energy

increases infinitely for infinitely large deformations. Mathematically speaking, the strain energy function Ψ must be a coercive function of the deformation \mathbf{F} . Second, a deformation \mathbf{F} which minimizes $\Psi = \Psi(\mathbf{F})$ exists, if, in addition to coercivity, Ψ is convex with respect to \mathbf{F} . However, if – in the case of compressing the body down to an infinitesimally small volume – the strain energy function increases infinitely (which makes sense from the physical view point), then it cannot be convex. Consequently, the convexity condition cannot be satisfied for the strain energy function.

In this context, the mathematical theory of solutions for the nonlinear elasticity problem mainly revolves around the concept of *polyconvexity* of strain energy functions Ψ . Particularly with respect to uniqueness and global existence of solutions, it is based on the polyconvexity condition of Ψ , see, e.g., the works of Ball [Bal77] or Ciarlet [Cia88]. According to John M. Ball, for isotropic hyperelasticity, a set of strain energy functions have been developed, which satisfy both the convexity and the coercivity condition [Bal77]. For anisotropic hyperelasticity, the search for ways to verify polyconvexity for a useful class of anisotropic stored-energy functions is still an open question and an area of active research even in the 21st century [Bal02].

Physically motivated, one finds that the elastic potential or strain energy function Ψ must satisfy specific invariance conditions: Firstly, known as the principle of *material objectivity*, Ψ should be invariant under superimposed rigid-body motions, i.e.

$$\Psi(\mathbf{F}) = \Psi(\mathbf{QF}) \quad (4.46)$$

for an arbitrary orthogonal tensor \mathbf{Q} (i.e., a rotation matrix). Hence, Ψ has to be independent from rotational components. Secondly, for reasons of simplicity, soft materials are often defined to be *isotropic*, meaning that their properties (e.g., the stress response) are identical in all directions, independent of any translation or rotation of the reference configuration. Mathematically, this is usually expressed in terms of the Cauchy-Green strain tensor \mathbf{C} :

$$\Psi(\mathbf{C}) = \Psi(\mathbf{QCQ}^T). \quad (4.47)$$

Using the *representation theorem of invariants* (see, e.g., [Hol00]), we can combine these requirements, and derive how strain energy functions Ψ for isotropic materials must be constructed. This theorem states that if a scalar-valued tensor function with the argument \mathbf{C} is an invariant under a rotation according to (eq. 4.47), it may be expressed in terms of the principal invariants of \mathbf{C} :

$$I_1(\mathbf{C}) = \text{tr}\mathbf{C}, \quad I_2(\mathbf{C}) = \frac{1}{2} \left[(\text{tr}\mathbf{C})^2 - \text{tr}(\mathbf{C}^2) \right], \quad I_3(\mathbf{C}) = \det\mathbf{C}. \quad (4.48)$$

The invariants $I(\mathbf{C}) = (I_1, I_2, I_3)$ are defined through the characteristic polynomial and hence related to the eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of \mathbf{C} .

Next, according to the Rivlin-Ericksen-Theorem (see, e.g., [CG07]), and assuming Ψ is continuously differentiable with respect to the invariants, one can expand Ψ into an infinite

power series

$$\Psi(I_1, I_2, I_3) = \sum_{p_1, p_2, p_3=0}^{\infty} c_{p_1, p_2, p_3} (I_1 - 3)^{p_1} (I_2 - 3)^{p_2} (I_3 - 1)^{p_3}, \quad (4.49)$$

where the coefficients c_{p_1, p_2, p_3} are material parameters that have to be determined by means of experiments.

Like this, several material models have been developed. The most common ones are listed in the following:

- *The Saint-Venant-Kirchhoff material model* is the simplest model for compressible hyperelastic materials. The strain energy function is usually formulated in terms of the Green-Lagrange strain tensor \mathbf{E} :

$$\Psi(\mathbf{E}) = \frac{\lambda}{2} (\text{tr} \mathbf{E})^2 + \mu \text{tr}(\mathbf{E}^2), \quad (4.50)$$

with the two Lamé parameters λ (*Lamé's first parameter*) and μ (*Lamé's second parameter* or *shear modulus*). In literature, the Lamé parameters are alternatively expressed in terms of other engineering constants, such as the *Poisson ratio* ν or the *Young's Modulus* E . For real materials, the following conditions for these parameters must hold: $E, \lambda, \mu > 0$ and $3\lambda + 2\mu > 0$.

The Saint-Venant-Kirchhoff model is often used for real-time applications in the computer graphics context. Based on the relationship of the work-conjugated stress-strain tensor pair \mathbf{S} and \mathbf{E} as in (eq. 4.42), we find the linear relation

$$\mathbf{S} = \frac{\partial \Psi}{\partial \mathbf{E}} = \lambda (\text{tr} \mathbf{E}) \mathbf{I} + 2\mu \mathbf{E}, \quad (4.51)$$

which is also known as the *linear material law of Hooke*. We remark that even though the stress-strain relation is linear, we are still dealing with *nonlinear elasticity*, since \mathbf{E} is nonlinear with respect to the displacement \mathbf{u} . Solving this equation numerically hence (still) requires iterative linearization methods like the Newton-Raphson-Method.

For small deformations the linear relation of the Saint-Venant-Kirchhoff material model approximates any arbitrary hyperelastic material model in first order. Although this model can be well-suited for many large displacement problems, too, its formulation has several disadvantages: In its development, it is not based on a decomposition of the strain energy function into a so-called isochoric and volumetric part, which entails that the third invariant $I_3 = \det \mathbf{E} = J$ (which is a measure for volumetric changes) is not even explicitly used. It is not monotonic in compression either, and can thus break down for large compressive strains. We remark that, consequently, it does not satisfy the above-mentioned polyconvexity condition. The *modified Saint-Venant-Kirchhoff material model* [Hol00] circumvents these serious drawbacks when used for large compressive strains.

- *The Mooney-Rivlin material model* was originally developed for incompressible, isotropic,

rubber-like materials and is often used for soft tissue modeling:

$$\Psi(\mathbf{C}) = c_1(I_1 - 3) + c_2(I_2 - 3). \quad (4.52)$$

This hyperelastic model approximates an incompressible material in second order.

- *The Neo-Hookean material model* is a special case of the Mooney-Rivlin material model, and generalizes Hooke's Law appropriately for large deformations:

$$\Psi(\mathbf{C}) = c_1(I_1 - 3), \quad (4.53)$$

where the material parameter c_1 can be associated with *shear modulus* μ by $\mu = 2c_1$.

4.1.3.3 Viscoelasticity and Damping

The stress response of a real biological soft tissue usually cannot be captured by purely hyperelastic models, since the behavior does not only depend on the instantaneous deformation and strain, but also on the deformation history. This can be observed, for instance, during simple indentation experiments. The material response during the loading phase is different from the unloading (*recovery*) phase. In particular, the stress will only gradually decrease over time (*relaxation*) after the indenter has been completely removed, which hyperelastic models cannot describe.

A general approach to model this *viscoelastic behavior* is the application of a time-dependent strain-energy function $\Psi(\mathbf{F}, t)$. Fung et al. [Fun93] therefore suggest to separate the purely hyperelastic material response from the viscoelastic behavior, known as the so-called Quasi-Linear-Viscoelasticity (QLV) approach. It is, however, considerably more computational intensive than a pure hyperelastic model. A more efficient, but less accurate approach is to use a *phenomenological viscosity formulation*, which is based on the computational model that we are going to deal with in Section 4.3.1.

If an elastic model is discretized in space using the Finite Element Method (see Section 4.3.1) and in time using a time integration method (see Section 4.3.2), the result in every time step is a system of ordinary differential equations (ODEs) with respect to the displacement:

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{f}_{\text{ext}}. \quad (4.54)$$

Here, \mathbf{U} is the so-called vector of nodal displacements, \mathbf{M} denotes the mass matrix, \mathbf{K} is the stiffness matrix, and \mathbf{f}_{ext} encapsulates the external forces. The idea of the phenomenologically motivated *Rayleigh Damping*, which is widely used to model internal structural damping, is to add an artificial damping term to the above equation

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{f}_{\text{ext}}. \quad (4.55)$$

In this formulation the damping matrix \mathbf{D} is constructed as a linear combination

$$\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K} \quad (4.56)$$

with the scalar coefficients α and β that control the specific viscoelastic behavior.

We will consider the mathematical theory of the above system of ODEs for linear elasticity below, and then particularly focus on the properties of the linear system and on the existence and uniqueness of a solution. At this point, above all, we emphasize the motivation of Rayleigh Damping with respect to stability: Similar to the hyperbolic wave equation, the solution of which is naturally undamped, the discrete solution of the above system (eq. 4.54) commonly shows unphysical oscillations. Here, adding a damping term then shows to have numerically stabilizing effects and lets this unphysical behavior vanish in the discrete case, which we will be dealing with below in Section 4.3. However, please be aware of the fact that the original elasticity model – as in (eq. 4.54) – is changed when a damping term is added: Physically seen, the added damping accounts for *energy dissipation* in dynamic systems, and affects that the (reversible) purely elastic behavior partly turns irreversible. Mathematically, one can thus achieve a transformation of the system matrix such that it obtains some of the nice properties of so-called *M-matrices* [CLL98], which are well-studied in connection with stability and convergence criteria for iterative methods for the solution of large sparse systems of linear equations. If for Δt and h (which denote time step size and finite element size, as will be introduced below) it holds that for $\Delta t, h \rightarrow 0$ the damping vanishes asymptotically, the model is consistent again [GR15]; see Section 4.3.2.

The generalization of the Rayleigh Damping scheme from linear elastic to nonlinear *hyperelastic* materials is straightforward [Hol00]. Basically, we then find that the discrete form (eq. 4.54) is not a linear system of ODEs anymore, but a nonlinear one. The time-discretization of the ODEs then yields a nonlinear system of equations, which is typically solved using the *Newton-Raphson algorithm*, i.e., by means of iteratively solving linear systems that originate from linearization of the nonlinear equations. Rayleigh Damping can be employed during this procedure, too, and then respectively appears in each linearization step.

4.1.3.4 Linear Elasticity

Solving a nonlinear system by means of the above mentioned Newton-Raphson algorithm (i.e., iterating over systems of linearized equations) is not only computationally expensive. Even worse, for dynamic problems it may lead to numerical instability.

Therefore, it is important to check if the nonlinear formulation can be replaced by an appropriate linearized one, which nevertheless adequately represents the physical behavior, too. This is, for instance, the case, when the deformation of the body is small, i.e., when $\|\nabla_R \mathbf{u}\| \ll 1$, and when the stress-strain relationship is linear, such as in (eq. 4.51) for Saint-Venant-Kirchhoff materials. Instead of solving the nonlinear problem, a computationally more efficient linear elasticity model can be used then.

Using a linear elasticity model reveals several advantages and disadvantages. Looking at run-time and performance, we will see in the following chapter(s), that linear elasticity serves as an important building block for real-time capable algorithms, and that it is also a lot simpler to implement, since the iterative nonlinear system linearization part of the Newton-Raphson algorithm is not needed anymore [Mez08].

We here list the linear elasticity model, and refer to standard literature, e.g., by Ogden [Ogd97], for a compelling derivation of the respective linear formulation from the above-presented nonlinear, hyperelastic model.

Conventionally, in the context of linear elasticity, it is more convenient to use the displace-

ment \mathbf{u} , see (eq. 4.3), as the primary variable. Also, as the reference configuration and the deformed configuration approximately coincide, the so-called spatial and material derivatives are the nearly the same. Consequently, the above-defined *infinitesimal Cauchy strain tensor*

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2} \left(\nabla_R \mathbf{u} + (\nabla_R \mathbf{u})^T \right) \approx \frac{1}{2} \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right) \quad (4.57)$$

can be alternatively expressed by means of the spatial derivative with respect to the current configuration. Under this approximation, the rate of deformation tensor

$$\dot{\boldsymbol{\varepsilon}} = \frac{1}{2} \left(\nabla \mathbf{v} + (\nabla \mathbf{v})^T \right) \approx \dot{\boldsymbol{\varepsilon}} \quad (4.58)$$

can be regarded as the time derivative of the infinitesimal Cauchy strain tensor $\dot{\boldsymbol{\varepsilon}}$.

Making use of this approximation, we find that under the small strain assumption in the framework of linear elasticity, $\dot{\boldsymbol{\varepsilon}}$ and the Cauchy stress tensor $\boldsymbol{\sigma}$ are work-conjugated, compare (eq. 4.42), and the strain energy rate per unit reference volume can be approximated by the strain energy rate per unit deformed volume:

$$\dot{\Psi} = \mathcal{W}_{\text{int}} = J \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}} \approx \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}}. \quad (4.59)$$

In order to obtain a material law for linear elasticity on the basis of the above approximation, we can subsequently proceed as above in the hyperelastic case. Having in mind that we want to achieve a *completely linear formulation*, we choose the linear stress-strain relation as in the Saint-Venant-Kirchhoff model (eq. 4.51). This yields

$$\boldsymbol{\sigma}(\mathbf{u}) = \frac{\partial \Psi(\mathbf{u})}{\partial \boldsymbol{\varepsilon}(\mathbf{u})} = \lambda(\text{tr} \boldsymbol{\varepsilon}(\mathbf{u})) \mathbf{I} + 2\mu \boldsymbol{\varepsilon}(\mathbf{u}) \quad (4.60)$$

for small strains. Note that this relation is now completely linear due to the linear stress-strain relationship and the linear dependency between the infinitesimal Cauchy strain $\boldsymbol{\varepsilon}$ and the displacement \mathbf{u} .

Also, we remark that when replacing the finite Green-Lagrange strain tensor \mathbf{E} in (eq. 4.51) by the infinitesimal Cauchy strain tensor $\boldsymbol{\varepsilon}$, we find that $\text{tr} \boldsymbol{\varepsilon}(\mathbf{u}) = \text{div} \mathbf{u}$, and – physically seen – the Lamé parameter λ then describes the stress resulting from changes in the material density.

At this point, we can state the complete form of the *(initial) boundary value problem for (fully-)linear elasticity* with respect to the displacement variable \mathbf{u} :

$$\begin{aligned} \text{Find } \mathbf{u} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N) \text{ such that} \\ \text{div}_R \boldsymbol{\sigma}(\mathbf{u}(\mathbf{x}_R)) + \rho_R \mathbf{g}(\mathbf{x}_R) = \rho_R \ddot{\mathbf{u}}(\mathbf{x}_R) \quad \forall \mathbf{x}_R \in \Omega_R, \end{aligned} \quad (4.61)$$

with $\boldsymbol{\sigma}$ as in (eq. 4.60), $\boldsymbol{\varepsilon}$ as in (eq. 4.57), and the respective function spaces

$$V_D(\Gamma_D, \mathbf{u}_D) := \{ \mathbf{u} | \mathbf{u}(\mathbf{x}_R) = \mathbf{u}_D \quad \forall \mathbf{x}_R \in \Gamma_D \}, \quad (4.62)$$

$$V_N(\Gamma_N, \mathbf{t}_N) := \{ \mathbf{u} | \boldsymbol{\sigma}(\mathbf{u}(\mathbf{x}_R)) \mathbf{n} = \mathbf{t}_N \quad \forall \mathbf{x}_R \in \Gamma_N \}. \quad (4.63)$$

In literature, this problem formulation is also known as the *Lamé-Navier (initial) boundary value problem of linear elasticity*.

We anticipate at this point, that the balance equation of linear elasticity in (eq. 4.61) is *elliptic* in the *stationary* case, i.e., when the acceleration $\ddot{\mathbf{u}} = \mathbf{0}$ for all $\mathbf{x} \in \Omega$. We will look at this again just below in the context of the theory on the variational formulation of linear elasticity in (eq. 4.78).

We again emphasize that the BVP (eq. 4.61) of linear elasticity is defined with respect to the reference configuration, although the Cauchy stress tensor $\boldsymbol{\sigma}$ is used instead of the first Piola-Kirchhoff tensor \mathbf{P} . As mentioned above, this approach is physically motivated by the circumstance that the reference configuration and the deformed configuration are nearly identical for small strains, which is why the Piola transformation then becomes unnecessary. Mathematically, for small strains, i.e., when $\|\nabla_R \mathbf{u}\| \ll 1$, quadratic terms in the finite Green-Lagrange strain tensor \mathbf{E} as in (eq. 4.51) can be neglected and one obtains the infinitesimal Cauchy strain tensor $\boldsymbol{\varepsilon}$. Furthermore, there exists a specific linear approximation of both the first Piola-Kirchhoff stress tensor and the second Piola-Kirchhoff stress tensor, which can be identified as the Cauchy stress tensor $\boldsymbol{\sigma}$, see Ogden [Ogd97].

For completeness, analogous to (eq. 4.41) and the respective derivation, we state the *balance equation of mechanical energy* for the *linear* elasticity formulation with respect to the reference configuration:

$$\frac{d}{dt} \int_{\Omega_R} \frac{1}{2} \rho_R \dot{\mathbf{u}}^2 dV_R + \int_{\Omega_R} \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}} dV_R = \int_{\partial\Omega_R} \mathbf{t} \cdot \dot{\mathbf{u}} d\mathbf{A}_R + \int_{\Omega_R} \rho_R \mathbf{g} \cdot \dot{\mathbf{u}} dV_R, \quad (4.64)$$

where we made use of $\mathbf{v} = \dot{\mathbf{x}} = \dot{\mathbf{u}}$. We remark that this formulation is an important result for our ongoing work towards using the Finite Element Method (FEM) for soft tissue simulation, see Section 4.3.1. We will present more details in the next paragraph on the theory of linear elasticity. Of course, the above described Rayleigh Damping can be used in the case of linear elasticity, too.

Theory of Linear Elasticity. As opposed to the hyperelastic case, for the above Lamé-Navier IBVP of linear elasticity (eq. 4.61), literature provides full theory on existence and regularity of solutions, see, e.g., Holzapfel [Hol00] or Rannacher [Ran08].

Starting from Hooke's linear material law (eq. 4.60), we define the fourth-order material tensor \mathcal{C} by $\boldsymbol{\sigma} = \mathcal{C} : \boldsymbol{\varepsilon}$, with $\mathcal{C} = (\mathcal{C}_{ijkl})_{i,j,k,l=1}^3$. It represents the above concept of the response function for linear elasticity, and is assumed to be *sufficiently smooth*, i.e., $\mathcal{C}_{ijkl} \in C^2(\bar{\Omega})$. Using Einstein's summation convention, we find

$$\sigma_{ij} = \mathcal{C}_{ijkl} \varepsilon_{kl}. \quad (4.65)$$

We assume that the body's material is isotropic, and keep in mind that, physically motivated (see above), the stress and strain tensors $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ are *symmetric*. This yields

$$\mathcal{C}_{ijkl} = \mathcal{C}_{jikl} = \mathcal{C}_{jilk}, \quad (4.66)$$

i.e., the symmetry of \mathcal{C} , see [Ran08]. Moreover, the body is assumed to be *homogeneous*, i.e., its physical properties are independent of the material point considered. This means that \mathcal{C} is

a tensor of elastic parameters that are constant:

$$\mathcal{C}(\mathbf{x}) = \mathcal{C} \quad \forall \mathbf{x} \in \Omega. \quad (4.67)$$

Finally, Rannacher [Ran08] proves that, for *isotropic* materials, the material tensor \mathcal{C} is *positive definite* if and only if the following holds for the material parameters:

$$\mu > 0, \quad 3\lambda + 2\mu > 0 \quad \text{or} \quad E > 0, \quad -1 < \nu < \frac{1}{2}, \quad (4.68)$$

respectively. We note that one will only need to require \mathcal{C} to be *positive definite on symmetric dyads*, however, without loss of generality, this does not imply the definiteness on all dyads. In particular, it cannot be expected that $\mathcal{C}\nabla\mathbf{u} : \nabla\mathbf{u} > 0$ for all $\nabla\mathbf{u} \neq 0$.

In the following, we will consider the IBVP of linear elasticity (eq. 4.61) from the *classical* point of view, meaning that we analyze the *existence* and *uniqueness* of classically differentiable solutions and their characteristic properties. We assume that the above mentioned properties of the material tensor \mathcal{C} hold, and additionally require that $\mathbf{g}, \mathbf{t}_N, \mathbf{u}_D$ are sufficiently smooth. This is considered and (if necessary) imposed in the simulation code, see below in Section 6.2. We then want to solve (eq. 4.61) such that for the components of $\mathbf{u} = (u_i)_{i=1}^3$, $\boldsymbol{\varepsilon} = (\varepsilon_{ij})_{i,j=1}^3$, $\boldsymbol{\sigma} = (\sigma_{ij})_{i,j=1}^3$ the following regularity conditions hold

$$u_i \in C(\Omega \cup \Gamma_D) \cap C^1(\Omega \cup \Gamma_N) \cap C^2(\Omega), \quad \varepsilon_{ij} \in C(\Omega) \cap L^2(\Omega), \quad \sigma_{ij} \in C(\Omega \cup \Gamma_N) \cap C^1(\Omega) \cap L^2(\Omega). \quad (4.69)$$

Under these assumptions, the *Theorem of Cosserat* then claims the uniqueness of the classical solution to (eq. 4.61), see [Ran08]:

Supposing that, \mathbf{u} and $\tilde{\mathbf{u}}$ denote two classical solutions to the BVP of linear elasticity (eq. 4.61), then there is a vector-valued function (or vector) $\mathbf{b} \in \mathbb{R}^3$ and a skew symmetric tensor (or matrix) $\mathbf{B} \in \mathbb{R}^{3 \times 3}$ such that

$$\mathbf{u}(\mathbf{x}) - \tilde{\mathbf{u}}(\mathbf{x}) = \mathbf{b} + \mathbf{B}\mathbf{x} \quad \forall \mathbf{x} \in \Omega, \quad (4.70)$$

which, as mentioned above, is a so-called *rigid body motion*. Additionally, if Γ_D contains three linearly independent vectors, we have $\mathbf{b} = \mathbf{0}$ and $\mathbf{B} = \mathbf{0}$ and the solution to (eq. 4.61) is unique.

We remark at this point that experiments with real soft tissue, as conducted, e.g., by Holzapfel [BNSH06] or Suwelack [Suw14], show that – in the Finite Element context (see Section 4.3.1) – a higher polynomial FE degree of 2 or 3 for \mathbf{u} yields better (i.e. more physically realistic) results than a polynomial FE degree of first order: Especially when modeling incompressible materials using low polynomial FE degrees, one may observe so-called *numerical locking phenomena*, which cause the body to behave a lot stiffer than it should, cf. [Ran08] and [Bra07]. Numerical locking can partly be mitigated or even avoided with higher polynomial FE degrees, see e.g. [CK92] or [BLME13]. From this finding, one may infer that the real solution is sufficiently smooth in order to claim higher regularity for \mathbf{u} than given in the regularity condition in (eq. 4.69), or, in the FEM context, respectively, to let $\mathbf{u} \in H_0^2(\Omega)^3$ (quadratic FE) instead of $\mathbf{u} \in H_0^1(\Omega)^3$ (linear FE), where $H_0^m(\Omega)$ denotes the so-called Sobolev spaces, which will be introduced in the following.

For the solution of the BVP of elasticity, we want to employ the *Finite Element Method* (FEM), which is one of the most powerful numerical discretization techniques for approximately solving initial and boundary value problems. It relies on a *variational formulation* of the BVP (eq. 4.61). In order to derive this formulation, we first introduce some important concepts along with the respective notation and definitions.

Let $L^2(\Omega)$ denote the Hilbert space of square-integrable scalar functions (in the Lebesgue sense) on some region Ω with the following scalar (or inner) product and norm:

$$(u, v)_\Omega = \int_\Omega uv \, dx, \quad \|u\|_\Omega = \sqrt{\int_\Omega |u|^2 \, dx}. \quad (4.71)$$

Generally, on $C^\infty(\bar{\Omega})$ we have the so-called *Sobolev norms of m -th order*, defined by

$$\|u\|_{m;\Omega} := \sqrt{\sum_{|\alpha| \leq m} \|D^\alpha u\|_\Omega^2}. \quad (4.72)$$

The completion of $C^\infty(\bar{\Omega})$ with respect to the norms $\|\cdot\|_{m;\Omega}$ then yields the *Sobolev spaces* $H^m(\Omega)$ for $m \in \mathbb{N}$ of weakly differentiable functions, and $L^2(\Omega)$ for $m = 0$, respectively, through equivalence class partitioning. A Sobolev space $H^m(\Omega)$ is also a Hilbert space (hence the notation $H^m(\Omega)$).

We denote by $\mathbf{L}^2(\Omega) := L^2(\Omega)^3$ and $\mathbf{H}^m(\Omega) := H^m(\Omega)^3$ the respective vector-valued counterparts of the before introduced spaces. In the subsequent analysis, we particularly refer to the Sobolev space $\mathbf{H}^1(\Omega) = H^1(\Omega)^3$, which is the completion of $C^\infty(\bar{\Omega})^3$ with respect to the norm given by

$$\|f\|_{1;\Omega} = \sqrt{\|f\|_\Omega^2 + \int_\Omega |\nabla f|^2 \, dx}, \quad (4.73)$$

and we let $\mathbf{H}_0^1(\Gamma_D; \Omega)$ be the closure of the subset

$$\{\mathbf{u} \in C^\infty(\bar{\Omega})^3 : \mathbf{u} = \mathbf{0} \text{ on } \Gamma_D\} \subseteq \mathbf{H}^1(\Omega). \quad (4.74)$$

Having introduced the concept of Sobolev spaces, we now want to derive a variational formulation of the BVP (eq. 4.61). There are two common ways for the derivation of this form: either transforming the BVP into the so-called *weak formulation* according to the standard approach as described in Braess [Bra07], or via the physics-based approach of minimizing certain energy or action functionals. We will briefly depict these two approaches below, and finally find that the two formulations are identical.

According to [Bra07], we start the derivation of a weak formulation from the BVP (eq. 4.61) by multiplication with an arbitrary *test function* $\mathbf{v} \in \mathbf{V}_0 := \mathbf{H}_0^1(\Gamma_D; \Omega)$ and integration over the volume of the body. This yields

$$- \int_{\Omega_R} \operatorname{div}_R \boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{v} \, dV_R + \int_{\Omega_R} \rho_R \ddot{\mathbf{u}} \cdot \mathbf{v} \, dV_R = \int_{\Omega_R} \rho_R \mathbf{g} \cdot \mathbf{v} \, dV_R, \quad (4.75)$$

with $\mathbf{u} \in \mathbf{V} := \{\mathbf{u} \in \mathbf{H}^1(\Omega) : \mathbf{u} = \mathbf{u}_D \text{ on } \Gamma_D\}$. The L^2 scalar product between the residual of the PDE (eq. 4.61) and the test functions \mathbf{v} hence is to vanish for all $\mathbf{v} \in \mathbf{V}_0 := \mathbf{H}_0^1(\Gamma_D; \Omega)$.

Like this, and via choosing Sobolev spaces for the solution and test function spaces, we hence introduce a new concept of a *solution*, a so-called *weak solution*. This will play a key role in the below section 4.3.1 when dealing with discretization in space by means of the Finite Element Method (FEM). Through this new concept, the PDE (eq. 4.61) is not required to hold *pointwise* for every point in the whole body Ω anymore, but rather only in the sense of the *integrated mean* with arbitrarily weighted test functions. The same also holds for the boundary conditions as given in (eq. 4.62) and (eq. 4.63). Moreover, as will be seen right below, partial integration and the choice of an appropriate test function space $\mathbf{H}_0^1(\Gamma_D; \Omega)$ which fits to the Dirichlet boundary condition will ensure that the two remaining boundary conditions in the BVP formulation are taken into account, too.

Following the standard derivation described in [Bra07], partial integration of the first term in (eq. 4.75) and use of the divergence theorems leads to the weak form of

$$\int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) : \nabla_R \mathbf{v} \, dV_R - \int_{\partial\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) \mathbf{n}_{\partial\Omega_R} \mathbf{v} \, dA_R + \int_{\Omega_R} \rho_R \ddot{\mathbf{u}} \mathbf{v} \, dV_R = \int_{\Omega_R} \rho_R \mathbf{g} \mathbf{v} \, dV_R, \quad (4.76)$$

and applying the boundary conditions as given in (4.62) and (4.63) we obtain

$$\begin{aligned} \int_{\Omega_R} \underbrace{\boldsymbol{\sigma}(\mathbf{u}) : \nabla_R \mathbf{v}}_{\stackrel{\text{sym.}}{=} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{v})} \, dV_R - \int_{\Gamma_N} \underbrace{\boldsymbol{\sigma}(\mathbf{u}) \mathbf{n}_{\Gamma_N}}_{=\mathbf{t}_N} \mathbf{v} \, dA_R - \underbrace{\int_{\Gamma_D} \boldsymbol{\sigma}(\mathbf{u}) \mathbf{n}_{\Gamma_D} \mathbf{v} \, dA_R}_{=0} + \int_{\Omega_R} \rho_R \ddot{\mathbf{u}} \mathbf{v} \, dV_R \\ = \int_{\Omega_R} \rho_R \mathbf{g} \mathbf{v} \, dV_R, \end{aligned} \quad (4.77)$$

where the third term equals to zero due to $\mathbf{v} \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$ by definition. This yields the *variational formulation* of the (instationary) IBVP (eq. 4.61) of linear elasticity:

Find $\mathbf{u} \in \mathbf{H}^1(\Omega)$ such that $\mathbf{u} - \mathbf{u}_D \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$ and

$$\int_{\Omega_R} \rho_R \ddot{\mathbf{u}} \mathbf{v} \, dV_R + \int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{v}) \, dV_R = \int_{\Gamma_N} \mathbf{t}_N \mathbf{v} \, dA_R + \int_{\Omega_R} \rho_R \mathbf{g} \mathbf{v} \, dV_R \quad (4.78)$$

$\forall \mathbf{v} \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$.

It is obvious, that each $\mathbf{u} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N)$, which solves the classical (instationary) IBVP (eq. 4.61) of linear elasticity, is also a solution to the above variational formulation in (eq. 4.78).

Vice versa, following Braess [Bra07], it can be shown that each $\mathbf{u} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N)$ for which (eq. 4.78) holds is also a solution of the original, classical PDE (eq. 4.61). On the other hand, (eq. 4.78) still makes sense if we assume *weaker regularity*, i.e., $\mathbf{u} \in V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N)$. Through the definition of weakly differentiable functions, the space of all solutions $\mathbf{u} \in V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N)$ to (eq. 4.78) can be identified with the Sobolev space $\mathbf{H}^1(\Omega)$.

At this point, the concept of weak solutions in Sobolev spaces is a valuable tool for the analysis of uniqueness and existence of solutions to the BVP: Using Korn's inequality one can show the H^1 -ellipticity of the stationary variational formulation, if Γ_D has a positive $(d - 1)$ -dimensional measure. Applying the Theorem of Lax-Milgram then proves that the problem is well-posed, i.e., that there exists a unique solution which depends continuously on the data. We additionally remark here, that the H^1 -ellipticity (as the essential precondition of the existence of a solution to the stationary variational formulation) also implies the stability of the *Galerkin solution* in the context of the Finite Element Method, see Section 4.3.1. Analogously, when the inertia term is unequal to zero, one finds that the instationary variational formulation is a hyperbolic equation, see, e.g., Evans [Eva10]. For further theoretical results with respect to existence and uniqueness of solutions of the instationary formulation (both with and without damping), we refer to standard literature [Eva10, Hol00].

With the fundamental lemma of the calculus of variations it can now be shown that the solution of the problem in the *strong* BVP form (eq. 4.61) is identical to the solution in the *weak* or *variational* form (eq. 4.78).

Looking upon the arbitrary test function \mathbf{v} in (eq. 4.78) as the *virtual displacement field*¹ δu , this leads to the fundamental *principle of virtual work*:

$$\int_{\Omega_R} \rho_R \ddot{\mathbf{u}} \delta u \, dV_R + \underbrace{\int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\delta u) \, dV_R}_{=\delta w_{\text{int}}(\mathbf{u}, \delta u)} = \underbrace{\int_{\Gamma_N} \mathbf{t}_N \cdot \delta u \, dA_R + \int_{\Omega_R} \rho_R \mathbf{g} \cdot \delta u \, dV_R}_{=\delta w_{\text{ext}}(\mathbf{u}, \delta u)}. \quad (4.79)$$

The principle of virtual work is the simplest variational principle, and obviously also an equivalent formulation of the above BVP of elasticity, given reduced regularity requirements. Physically interpreted, the principle of virtual work states: the virtual stress work at fixed $\boldsymbol{\sigma}(\mathbf{u})$ together with the virtual kinematic work along δu is equal to the work done by the body force $\rho_R \mathbf{g}$ per unit current volume and the surface traction \mathbf{t} per unit current surface along δu , see [Hol00].

The functions δw_{int} and δw_{ext} are known as the *internal (mechanical) virtual work* and the *external (mechanical) virtual work* in mechanics. In the first case, the stress $\boldsymbol{\sigma}(\mathbf{u})$ does the internal work along the virtual displacements δu (and their resulting virtual strains $\delta \boldsymbol{\varepsilon}(u)$). In the second case, external work is done by the loads, which are the body force and the surface traction, along the virtual displacement δu on the region Ω and its boundary surface $\partial\Omega$, respectively. For vanishing accelerations $\ddot{\mathbf{u}}$, the internal virtual work equals the external virtual work, i.e., $\delta w_{\text{int}} = \delta w_{\text{ext}}$. For further details, see, e.g., the books by Marsden and Hughes [MH83] or Holzapfel [Hol00].

¹*The concept of virtual displacements:* We consider two neighboring displacement fields u and $\tilde{u} = u + \varepsilon w$, where w is an entirely new vector field which yields a virtual, slightly modified configuration in the neighborhood of u , with a scalar parameter ε . Following *Lagrange*, we call the difference between two neighboring displacement fields $\delta u = \tilde{u} - u = \varepsilon w$ the (first) variation δu of the displacement field u . In mechanics it is also known as the *virtual displacement field*. The variation of u is assumed to be an *arbitrary, infinitesimal* (since $\varepsilon \rightarrow 0$) and a *virtual* change, i.e., an imaginary (not a *real*) change. Note that, opposed to this, du also characterizes an *infinitesimal change* of u , however, du refers to an *actual* change. For additional information on the area of variational principles and variational calculus in general, we refer to [Hol00] or [TT60].

The second (alternative) approach for deriving the variational formulation goes via the minimization of energy or action functionals. In the stationary (time-independent) case, the variational formulation is derived from the balance of mechanical energy (eq. 4.64) via the definition of the total potential energy of a physical system and its minimization. This physics-based approach thus provides a very intuitive and elegant access to the weak formulation. In the instationary case, the intuitive derivation via the minimization of the elastic energy is replaced by the *principle of least action* [Ibr09], which leads to the search for a stationary point of the so-called Hamiltonian variational principle. For a detailed derivation and analysis, we refer to standard literature, e.g., Holzapfel [Hol00].

Here, we only look at the case of *time-independent (stationary) deformations*. We start by defining the *total potential energy* $E_{\text{pot}}(\mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon})$ of a physical system, which is obtained from (eq. 4.64) by omitting the inertia term and integrating over time:

$$\begin{aligned} E_{\text{pot}}(\mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon}) &:= \frac{1}{2} \int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{u}) \, dV_R - \int_{\Gamma_N} \mathbf{t}_N \mathbf{u} \, dA_R - \int_{\Omega_R} \rho_R \mathbf{g} \mathbf{u} \, dV_R \\ &= \frac{1}{2} w_{\text{int}}(\mathbf{u}, \mathbf{u}) - w_{\text{ext}}(\mathbf{u}, \mathbf{u}). \end{aligned} \quad (4.80)$$

According to the *principle of the minimum of the potential energy*, we then find that a kinematically admissible state $S = \{\mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon}\}$ of a dynamic system is solution to the BVP of linear elasticity (eq. 4.61), if and only if it minimizes the potential energy $E_{\text{pot}}(\mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon})$ over all kinematically admissible states $\tilde{S} = \{\tilde{\mathbf{u}}, \tilde{\boldsymbol{\sigma}}, \tilde{\boldsymbol{\varepsilon}}\}$, i.e.:

$$E_{\text{pot}}(S) < E_{\text{pot}}(\tilde{S}) \quad \forall \tilde{S}. \quad (4.81)$$

Hence, carrying out the variation on the total potential energy $E_{\text{pot}}(\mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon})$ to find a solution to the minimization problem in (eq. 4.81), then yields an equivalent formulation of the variational formulation of linear elasticity as in (eq. 4.78).

The principle of the minimum of the potential energy goes back to the ideas of Green (1839) and Kirchhoff (1850), and was mathematically founded for the first time by Hadamard (1903) and Love (1906). Physically interpreted, it states that for the actual solution of the BVP, the difference of internal stress energy (i.e., the stored energy in the deformed body) and the work done by the external forces (i.e., the loss of energy during the deformation process) is minimal as compared to all other kinematically admissible states.

The principle holds for functions $(\mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\varepsilon})$ that are kinematically admissible, i.e., for which the balance equation (eq. 4.61) along with the boundary conditions as in (4.62) and (4.63) is not violated. We remark that, besides the above *primary principle*, alternative formulations consider *statically* admissible functions in the context of so-called *dual principles*, and that there are also mixed *primary-dual formulations*, e.g., by Hellinger and Reissner, see [Bra07].

Summarizing, according to the theorem of Cosserat and to the principle of the minimum of the potential energy, classical solutions to the BVP of linear elasticity (eq. 4.61) are unique and can be characterized by means of their extremal properties. Following Rannacher [Ran08], in order to analyze the existence of such solutions, we require the subsequent assumptions (A1) to (A4) to hold:

- (A1) The subset $\Omega \subset \mathbb{R}^3$ satisfies the strong cone condition.
- (A2) The measure of $\Gamma_D \subset \partial\Omega$ is positive and its relative interior contains three linearly independent vectors.
- (A3) On Ω , the stiffness tensor $C = (C_{ijkl})$ is symmetric and uniformly bounded,

$$C_{ijkl} = C_{klij} \in L^\infty(\Omega), \quad (4.82)$$

as well as positive definite, i.e.,

$$C_{ijkl}\varepsilon_{ij}\varepsilon_{kl} > 0 \quad (4.83)$$

for all symmetric tensors $\varepsilon \neq 0$.

- (A4) The external forces acting on the solid and the displacement of the boundary are given by

$$\mathbf{g} \in L^2(\Omega)^3, \quad \boldsymbol{\sigma}_N \in H^1(\partial\Omega)_{\text{sym}}^{3 \times 3}, \quad \mathbf{u}_D \in H^1(\Omega)^3. \quad (4.84)$$

We remark at this point, that the data and model we use (i.e., the geometric information about the tissue and the body, the material parameters, the boundary conditions, etc.) do satisfy these conditions, either intrinsically or through imposing according constraints, see Section 6.2.

Given that the above assumptions (A1) to (A4) hold, the bilinear form

$$a(\mathbf{u}, \mathbf{v}) := \int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{v}) \, dV_R \quad (4.85)$$

and the linear form

$$l(\mathbf{v}) := \int_{\Gamma_N} \mathbf{t}_N \mathbf{v} \, dA_R + \int_{\Omega_R} \rho_R \mathbf{g} \mathbf{v} \, dV_R \quad (4.86)$$

are well-defined on the Hilbert space $H^1(\Omega)^3$, as well as bounded and continuous.

In the definition of l , we do not consider the given Dirichlet boundary conditions as in (4.62). And through the choice of the test function space $\mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$, we preliminarily obtain a formulation as if we considered homogeneous Dirichlet boundary conditions, i.e., $\mathbf{u}_D = \mathbf{0}$ on Γ_D . We will correct this error by requiring the difference $\mathbf{u} - \mathbf{u}_D$ of the solution \mathbf{u} and the given vector-valued translation field \mathbf{u}_D to vanish on the boundary Γ_D , i.e., $\mathbf{u} - \mathbf{u}_D \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$. Of course, we also need to take care of this in the implementation of the soft tissue simulation, where the Dirichlet boundary conditions are considered separately in the assembly of the linear system, see Sections 4.3.2.3 and 5.2.1. The resulting variational formulation of the equilibrium equations of linear elasticity in the stationary case now reads as follows:

Find $\mathbf{u} \in \mathbf{H}^1(\Omega)$ such that $\mathbf{u} - \mathbf{u}_D \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$ and

$$a(\mathbf{u}, \varphi) = l(\varphi) \quad \forall \varphi \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega). \quad (4.87)$$

with $a(\cdot, \cdot)$ and $l(\cdot)$ as defined in (eq. 4.85) and (eq. 4.86).

Given that (A1) to (A4) hold, according to Rannacher [Ran08], the *existence theorem* (Theorem 1.10) states that there exists a unique weak solution to the equilibrium equations of linear elasticity. This weak solution \mathbf{u} is a strict minimum of the energy functional

$$E(\mathbf{u}) := \frac{1}{2}a(\mathbf{u}, \mathbf{u}) - l(\mathbf{u}) \quad (4.88)$$

on $\mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega)$.

According to the derivation of the weak formulation of the problem of linear elasticity, every classical solution $\mathbf{u} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N)$ is also a weak solution $\mathbf{u} \in \mathbf{V} = \{\mathbf{u} \in \mathbf{H}^1(\Omega) : \mathbf{u} = \mathbf{u}_D \text{ on } \Gamma_D\}$. Vice versa, given the assumptions (A1) to (A4) hold, the *regularity theorem* (Theorem 1.11) in [Ran08] states that if Γ_D and Γ_N are 2-dimensional $C^{2+\alpha}$ -submanifolds, and if the regularity conditions

$$\mathcal{C} \in C^2(\bar{\Omega})^{3^4}, \quad \mathbf{g} \in C^\alpha(\bar{\Omega})^3, \quad \boldsymbol{\sigma}_N \in C^{1+\alpha}(\bar{\Omega})_{\text{sym}}^{3 \times 3}, \quad \mathbf{u}_D \in C^{2+\alpha}(\bar{\Omega})^3 \quad (4.89)$$

are satisfied, the weak solution is a solution in the classical sense with respect to the original BVP of linear elasticity (eq. 4.61), too, i.e.,

$$\mathbf{u} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N). \quad (4.90)$$

This corresponds to the remarks given after the box of (eq. 4.78). For a proof of this theorem, we refer to the textbook by Ciarlet [Cia88].

With respect to the implementation of our linear elasticity-based soft tissue simulation (see Section 5.2.1), we point out that one can directly recognize the terms of the above variational formulation (respectively see equations 4.85 and 4.86) in the assembly formulation.

Closing, we remark that the solution generally exhibits a certain singular behavior at the boundary interface regions where Dirichlet boundaries get to meet effective Neumann boundaries. This is due to the potentially unbounded gradient or stress tensor, respectively. In order to prevent this problem, it is recommended to have a *layer of zero Neumann boundary* between the Dirichlet boundary region and the effective Neumann boundary region. Note that this also holds for contact boundary regions (see Section 4.1.4) if these are handled by means of a penalty term that is represented as a penalizing pressure condition, i.e., in a way similar to a Neumann pressure boundary condition.

Summarizing, the theory presented shows the existence, uniqueness and regularity of the solution \mathbf{u} to the BVP of linear elasticity (eq. 4.61). For more details on derivation and proofs, we refer to [Ran08] and [Cia88].

We remark that in this work, from now on, we assume that the above listed requirements hold, and we will consider *all* soft bodies to be *homogeneous, objective, isotropic, and linear-elastic*. Yet, it is important to point out that the linear model, as it is based on the small strain assumption, usually is not justified for soft tissue deformations (and rotations), which generally may be arbitrarily large. As discussed above and in the State-of-the-Art Section, the material response of cardiac soft tissue is highly nonlinear.

It is hence to be expected that the linear elasticity model introduces a significant error in soft tissue simulations. Particularly, when deformations become significantly large, the linearization

of the strain tensor does not produce satisfying results anymore, and so-called *ghost forces* cause the deformed object to blow up unnaturally [SSS⁺13, Mez08, MMD⁺02]. A simple test scenario, where the bending of a beam is simulated using a linear elasticity formulation, shows the effects of these ghost forces; see left column in Figure 4.3. Hence, for large deformations, the usage of more sophisticated elasticity models is required. However, these more evolved elasticity models often are too computationally expensive, especially when aiming for real-time soft tissue simulations. For example, the physically correct nonlinear formulation requires repeated linearization by means of, e.g., the Newton-Raphson algorithm as mentioned above, and hardly delivers complex simulation results in real-time.

A smart approach to overcome these issues originates from research in the context of Computer Graphics: the so-called *corotation-based approach*. We will briefly depict this approach in the subsequent paragraph.

4.1.3.5 An Alternative Formulation: The Corotation-based Approach

An elegant solution to the problem of finding a compromise over obtaining both highly accurate and realistic solutions, while still only paying low computational costs, is the so-called corotation-based elasticity formulation. Via polar decomposition, it treats separately the *rigid rotation* and the *actual deformation* of a body (and its elements, respectively) while still using the linear Cauchy strain tensor.

We particularly point out the works of Bro-Nielsen [BN98], who reviews the application of corotation-based FEM simulations in the context of Virtual Reality in medical applications, or Mezger et al [MTPS08], who analyzes the corotational formulation for interactive physically-based shape editing. When combined with the corotation-based approach, an application of the infinitesimal strain theory of linear elasticity yields stable and realistic results even for large soft tissue deformations with only small computational efforts [SSS⁺13], see right column in Figure 4.3. From the computation point of view, the corotation-based formulation can even be enhanced, e.g., with respect to the frequency of stiffness matrix updates, known as *lazy corotation*, or by means of a specific element rotation matrix assembly, see [GW08] and [HS90].

Extensive numerical validation and experiments have shown that corotational simulation of deformable solids makes linear elasticity consistent with the (nonlinear) theory of hyperelasticity, not only for small deformations but also for larger ones [HS04, HGS03]. Going further, Suwelack [Suw14] successfully validated the combination of the corotation-based formulation along with isoparametric quadratic tetrahedra, and its application for soft tissue modeling in the context of liver surgery assistance. It is also to be noted that, as a consequence, this consistency is often used to determine some of the parameters of hyperelastic material models: the consistency conditions can easily be found by comparing Hooke's law with linearized hyperelasticity at small strains.

Of course, one can consider more complicated constitutive equations, which can also be of inelastic nature. We refer to [SH98] or [Hol00] for more information.

Again, we emphasize that we assume *all* soft bodies to be *homogeneous, objective, isotropic, and linear-elastic* in this work, and that our prototypic soft tissue simulation for surgery assistance is implemented accordingly; see Section 4.3.1 and Chapter 5.

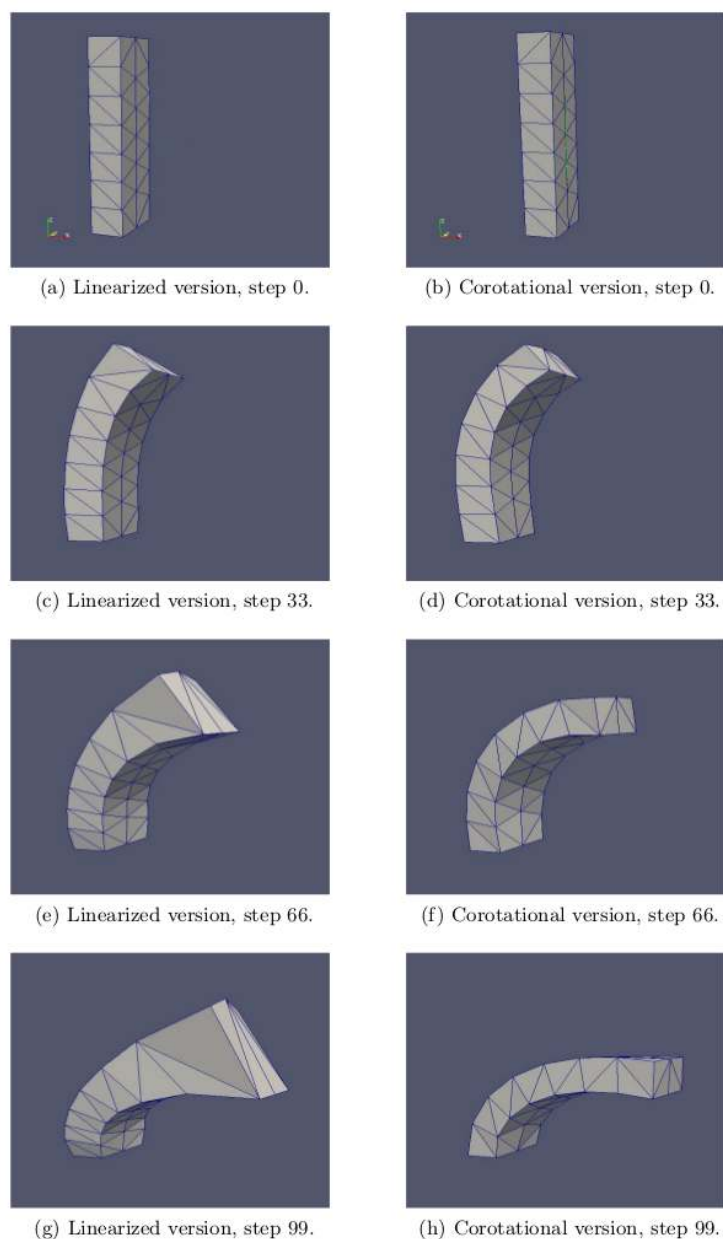


Figure 4.3: Visualization of a simple soft tissue manipulation experiment based on two different elasticity formulations: *linear elasticity* in the left column and *corotation-based elasticity* in the right column. The material parameters and boundary conditions are the same in both models: Over a 10 sec period which is discretized by 100 time steps, a beam that consists of 100 linear-tetrahedral elements (Tet4 elements) is fixed at the bottom and bended at the upper right side via prescribed displacement conditions. In the linear elastic case, *ghost forces* occur and cause the body to unnaturally blow up, hence revealing an unrealistic behavior. In contrast, the corotation-based formulation shows physically realistic results that coincide with simulation outcomes that are based on nonlinear elasticity, cf. [SSS⁺13, Mez08].

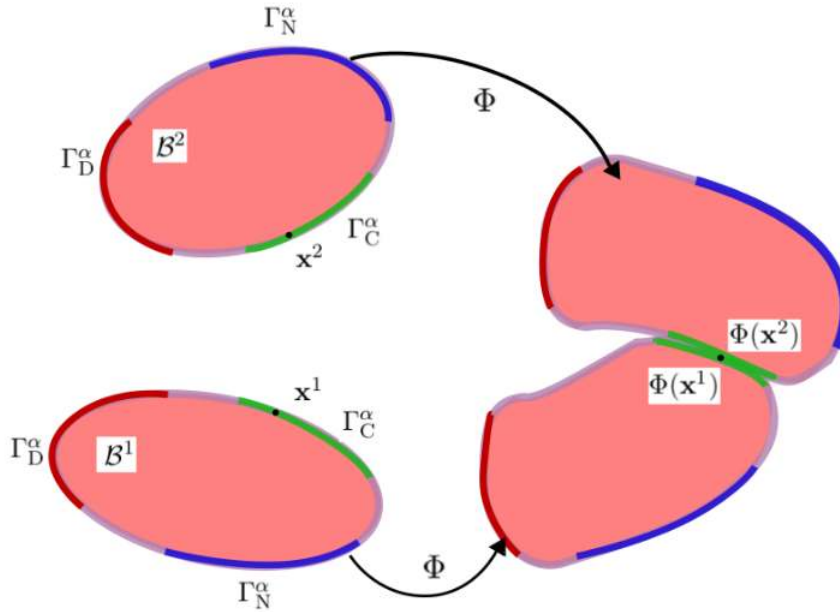


Figure 4.4: Schematic draft of the deformation of two soft bodies that get into contact, along with the respective notation explained in the text.

4.1.4 Contact Boundary Value Problem and Variational Form

So far, using the above general *Lamé-Navier IBVP formulation of elasticity* – see again (eq. 4.31) and (eq. 4.32), or for the linear elastic case (eq. 4.61) along with the initial conditions – we can model the free deformation of an elastic body subjected to the respective boundary conditions. However, we do not yet consider the behavior of two bodies or two body parts, that get in touch with each other (*contact*) and possibly cause mutual deformation. In order to do so, additional assumptions need to be made, and according constraints must be defined. We therein closely abide to the setup suggested in the textbook by Wriggers [Wri06].

Generally, contact problems in biomechanics can involve large deformations of the bodies that are in contact. Therefore, contact relations are usually formulated for finite deformations. In this work, however, we restrict the derivation of the respective formulation to the case of linear elasticity for reasons of simplicity. We look at problems where two bodies approach each other during a deformation process and come into contact on parts of their boundaries denoted by Γ_C , see Figure 4.4. Obviously, we can observe that two points \mathbf{x}^1 and \mathbf{x}^2 , that are distinct in the initial configuration of the bodies, may occupy the same position in the current configuration, i.e., $\Phi(\mathbf{x}^1) = \Phi(\mathbf{x}^2)$, within the deformation process. Hence, contact conditions need to be formulated with respect to the current configuration. This is particularly important for linear elasticity, where the formulation needs to be re-transformed for the computation, see below in this paragraph and in the FEM section 4.3.1.

In general, two steps have to be performed to detect whether or not contact takes place. These are the global search for contact and the setup of local kinematical relations which are needed to formulate the contact constraints. The search for contact will be depicted later, in

Section 5.2.2. At this point, we will focus on the formulation of appropriate contact constraints.

In general, one can distinguish two cases: On the one hand, there is the *small penetration theory*, where small penetrations between the two bodies in contact are allowed; see, e.g., the work by Wriggers and Miehe in 1992 [WM92]. On the other hand, one finds the *classical non-penetration theory*, where the distance between two bodies in contact is minimized; see, e.g., the work of Curnier and Alart in 1988 [CA88]. The latter again differentiates between *normal contact*, which generally ensures that none of the bodies is interpenetrated by the other one, and *tangential contact*, which on top of this allows to formulate frictional problems, where the two bodies may either stick to each other at certain points, or where they slide along each other in the contact interface, see, e.g., [SL92]. There exist specific mathematical formulations for all cases. For more details, we refer to the suggested literature.

In this work, we will assume *frictionless contact*, and start by considering two elastic bodies \mathcal{B}^α , $\alpha = 1, 2$, each occupying the bounded domain $\Omega^\alpha \subset \mathbb{R}^3$, see Figure 4.4. The boundary Γ^α of a body \mathcal{B}^α consists of three parts: besides the already known Γ_D^α with prescribed displacements and Γ_N^α with prescribed surface loads, we now additionally have $\Gamma_C^\alpha \subseteq \Gamma_N^\alpha$, where the two bodies \mathcal{B}^1 and \mathcal{B}^2 come into contact. In the context of our intended contact simulation, where we do not precisely know in advance which parts of the boundary actually do come into contact, we additionally define a so-called *potential contact boundary* $\Gamma_{C_{\text{pot}}}^\alpha$, with $\Gamma_C^\alpha \subseteq \Gamma_{C_{\text{pot}}}^\alpha$. We require $\Gamma_C^\alpha \cap \Gamma_N^{\neq 0; \alpha} = \emptyset$ for every instant of time. In the following, we will omit the α when obvious from the context.

For the formulation of the BVP, we only have to discuss the additional terms due to contact in detail. The equations describing the behavior of the two bodies coming into contact do not change. We therefore refer to (eq. 4.61) for the linear elastic case and will subsequently formulate adequate additional contact constraints in the contact area. Finally, since the FEM will be applied to solve the resulting BVP numerically, we again have to derive the variational formulation for the respective contact BVP.

Note that later in this work the implied two bodies will actually represent *two body regions* (the two mitral valve leaflets) of a *single elastic body* (the mitral valve); each of these body regions is annotated with another ID.

Using the Euclidean distance between two arbitrary points \mathbf{a} and \mathbf{b}

$$d(\mathbf{a}, \mathbf{b}) := \|\mathbf{a} - \mathbf{b}\|_2, \quad (4.91)$$

we define the *gap distance* of a body's boundary point \mathbf{x}^1 in the current configuration (i.e., $\Phi(\mathbf{x}^1)$) as its distance to the closest point $\Phi(\mathbf{x}^2)$ on the other body:

$$d_{\text{gap}}(\mathbf{x}^1) := \min_{\mathbf{x}^2 \in \Gamma_C^2} d(\Phi(\mathbf{x}^1), \Phi(\mathbf{x}^2)) = \min_{\mathbf{x}^2 \in \Gamma_C^2} \|\Phi(\mathbf{x}^1) - \Phi(\mathbf{x}^2)\|_2. \quad (4.92)$$

Obviously, in order to guarantee that none of the bodies interpenetrates the other one, d_{gap} must always satisfy

$$d_{\text{gap}}(\mathbf{x}) \geq 0. \quad (4.93)$$

With the definition of the *contact pressure*

$$p_C(\mathbf{x}) := \mathbf{n}_{\Gamma_C} \boldsymbol{\sigma}(\mathbf{x}) \mathbf{n}_{\Gamma_C} \quad (4.94)$$

with the surface normal vector \mathbf{n}_{Γ_C} on the contact boundary Γ_C , we let

$$p_C(\mathbf{x}) \geq 0 \quad (4.95)$$

for all boundary points at all times, which makes sure that no boundary point is *caught* at the other body's boundary.

Following Wriggers [Wri06], we impose that one of the two inequalities (eq. 4.93) or (eq. 4.95) is satisfied as an equality by letting

$$d_{\text{gap}}(\mathbf{x}) p_C(\mathbf{x}) = 0. \quad (4.96)$$

Considered pointwise, this means that either (a) a point $\Phi(\mathbf{x}^1)$ on the one boundary is still separated from the any point $\Phi(\mathbf{x}^2)$ on the other body's boundary (i.e., $d_{\text{gap}}(\mathbf{x}^1) > 0$) and the contact pressure $p_C(\mathbf{x}^1)$ is zero, or (b) the contact pressure is unequal to zero, and the body's boundary point $\Phi(\mathbf{x}^1)$ is in contact with some point $\Phi(\mathbf{x}^2)$ on the other body's boundary (i.e., $d_{\text{gap}}(\mathbf{x}^1) = 0$), see again Figure 4.4.

Practically, being aware of the upcoming discretization in space and time (see Sections 4.3.1 and 4.3.2), which is needed to numerically solve the contact problem of elasticity, we modify the above conditions slightly in order to prevent the discrete representations of the two bodies from interpenetrating each other within two subsequent discrete time steps t_n and t_{n+1} .

Firstly, we replace the 0 in (eq. 4.93) by some $\varepsilon_{\text{tol}} > 0$, which denotes the minimally admissible threshold gap distance, below which the two bodies are *defined to be in contact*. Secondly, for $0 < d_{\text{gap}} < \varepsilon_{\text{tol}}$, we let $p_C(\mathbf{x}) := p_C(\mathbf{x}, d_{\text{gap}}(\mathbf{x}))$ increase with decreasing gap distance:

$$p_C(\mathbf{x}) := p_C(\mathbf{x}, d_{\text{gap}}(\mathbf{x})) = \frac{\varepsilon_{\text{tol}}}{d_{\text{gap}}(\mathbf{x})} - 1. \quad (4.97)$$

We thus obtain the following set of conditions for modeling contact:

$$\begin{aligned} d_{\text{gap}}(\mathbf{x}) &\geq \varepsilon_{\text{tol}}, \\ p_C(\mathbf{x}, d_{\text{gap}}(\mathbf{x})) &\geq 0, \\ (d_{\text{gap}}(\mathbf{x}) - \varepsilon_{\text{tol}}) p_C(\mathbf{x}, d_{\text{gap}}(\mathbf{x})) &= 0. \end{aligned} \quad (4.98)$$

Incorporation of these conditions within an appropriate functional leads to a category of mathematical problems of finding the constrained minimum of the appropriate functional, where d_{gap} is the constraint condition and normal forces p_C can be recognized as *Lagrange multipliers*. A discrete version of the Lagrange multiplier method for 3D contact problems with friction was employed by Chaudhary and Bathe [CB86]. The kinematical constraint (eq. 4.93) is exactly satisfied in this method. However, apart from an extended number of unknowns, this approach can cause difficulties in the solution phase due to the occurrence of zeros on the diagonal of the associated algebraic system.

By choosing the *penalty method*, the constraint condition (eq. 4.93) is relaxed, and we can again apply a classical displacement-based BVP formulation as in (eq. 4.61), cf. [PO92]. In addition to the standard formulation of the BVP of linear elasticity (eq. 4.61) we thus require the following contact boundary condition to hold:

$$\boldsymbol{\sigma}(\mathbf{u}(\mathbf{x})) \mathbf{n} = \mathbf{p}_C \quad \text{on } \Gamma_C, \quad \text{where } \mathbf{p}_C = \begin{cases} \mathbf{0} & \text{if } d_{\text{gap}}(\mathbf{x}) > \varepsilon_{\text{tol}} \text{ on } \Gamma_C, \\ p_C(\mathbf{x}, d_{\text{gap}}(\mathbf{x})) \mathbf{n}_{\Gamma_C} & \text{if } d_{\text{gap}}(\mathbf{x}) \leq \varepsilon_{\text{tol}} \text{ on } \Gamma_C. \end{cases} \quad (4.99)$$

Please note that the penalty term \mathbf{p}_C is defined on a part of the (contact) boundary *surface* (later: a surface element), however, it is scaled with respect to a *pointwise* gap distance d_{gap} of *one* representative point on this surface part (later: a boundary point in the center of the respective surface element).

Summarizing, we obtain the *contact BVP formulation for linear elasticity* with respect to the displacement variable \mathbf{u} :

$$\begin{aligned} \text{Find } \mathbf{u} \in C^2(\Omega)^3 \cap C^1(\bar{\Omega})^3 \cap V_D(\Gamma_D, \mathbf{u}_D) \cap V_N(\Gamma_N, \mathbf{t}_N) \cap V_C(\Gamma_C, \mathbf{p}_C) \text{ s.t.} \\ \operatorname{div}_R \boldsymbol{\sigma}(\mathbf{u}(\mathbf{x}_R)) + \rho_R \mathbf{g}(\mathbf{x}_R) = \rho_R \ddot{\mathbf{u}}(\mathbf{x}_R) \quad \forall \mathbf{x}_R \in \Omega_R, \end{aligned} \quad (4.100)$$

with $\boldsymbol{\sigma}$ as in (eq. 4.60), ε as in (eq. 4.57), and the respective function spaces

$$V_D(\Gamma_D, \mathbf{u}_D) := \{\mathbf{u} \mid \mathbf{u}(\mathbf{x}_R) = \mathbf{u}_D \quad \forall \mathbf{x}_R \in \Gamma_D\}, \quad (4.101)$$

$$V_N(\Gamma_N, \mathbf{t}_N) := \{\mathbf{u} \mid \boldsymbol{\sigma}(\mathbf{u}(\mathbf{x}_R)) \mathbf{n} = \mathbf{t}_N \quad \forall \mathbf{x}_R \in \Gamma_N\}, \quad (4.102)$$

$$V_C(\Gamma_C, \mathbf{p}_C) := \{\mathbf{u} \mid \boldsymbol{\sigma}(\mathbf{u}(\mathbf{x}_R)) \mathbf{n} = \mathbf{p}_C \quad \forall \mathbf{x}_R \in \Gamma_C\}. \quad (4.103)$$

Finally, to derive the *variational formulation* for the (instationary) *contact BVP of linear elasticity*, we proceed according to the standard derivation as in the above subsections, such that – analogously to (4.78) – we obtain

$$\begin{aligned} \text{Find } \mathbf{u} \in \mathbf{H}^1(\Omega) \text{ such that } \mathbf{u} - \mathbf{u}_D \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega) \text{ and} \\ \underbrace{\int_{\Omega_R} \rho_R \ddot{\mathbf{u}} \mathbf{v} dV_R}_{\text{inertia}} + \underbrace{\int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}) : \boldsymbol{\varepsilon}(\mathbf{v}) dV_R}_{\text{stiffness}} \\ = \underbrace{\int_{\Omega_R} \rho_R \mathbf{g} \mathbf{v} dV_R}_{\text{body force}} + \underbrace{\int_{\Gamma_N} \mathbf{t}_N \mathbf{v} dA_R}_{\text{pressure}} + \underbrace{\int_{\Gamma_C} \mathbf{p}_C \mathbf{v} dA_R}_{\text{contact}}, \quad (4.104) \\ \forall \mathbf{v} \in \mathbf{V}_0 = \mathbf{H}_0^1(\Gamma_D; \Omega). \end{aligned}$$

with the additional term for Γ_C in order to consider the event of contact. Of course, damping (as suggested above) can be introduced here, too. We remind the reader that $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ is actually space and time dependent, and that the formulation is space and time continuous.

Besides the above, there are alternative ways to model and simulate elastic contact. Wriggers [Wri06] provides an extensive overview of the most common methods. The bigger part of these methods share that the above contact constraints setup (eq. 4.98) is described in a displacement-based formulation, and such that the emerging BVP includes the inequalities. This is the so-called *Signorini's problem*, adapted from A. Signorini, who was one of the first to investigate on the problem class of contact in elasticity.

Aiming for the application of the FEM, the derivation of the variational formulation leads to a *variational inequality*. This is different from the usual situation in solid mechanics, where the solutions have to fulfill variational equations. Due to the inequality constraint on the deformation field, the contact problem is *nonlinear* even in the case of linear elasticity, which requires special algorithms to be designed for the upcoming *nonlinear optimization problem*. Among these algorithms, the most prominent classes are the so-called *active set strategies*, which are applied in combination with *Lagrange multipliers*, and the *penalty techniques*. Both of these algorithm classes and their respective sub-classes are designed to fulfill the constraint equations in the normal direction in the contact interface and hence work well for normal contact. For the tangential part, special constitutive relations need additionally to be considered when stick and slip motion occurs at the contact interface; associated techniques are discussed, e.g., in Wriggers [Wri06]. For the specific advantages and disadvantages of the different methods, we refer to the literature. Here, we only mention the Mortar-element-based approach by Wohlmuth and colleagues, who successfully adopted mortar finite elements to variational inequalities originating from contact problems, and achieved stable discretizations even for nonmatching grids, see, e.g., [WK03].

Looking at the penalty method and the above derived formulation again, it is clear that the penalty-based solution of (eq. 4.104), with \mathbf{p}_C according to (eq. 4.99) and to (eq. 4.97), is only an approximation of the correct enforcement of the constraint condition, which may, e.g., be obtained by means of the Lagrange multiplier approach. In the limit $p_C(\mathbf{x}, d_{\text{gap}}(\mathbf{x})) \rightarrow \infty$ for $d_{\text{gap}}(\mathbf{x}) \rightarrow 0$ as in (eq. 4.97), the penalty method theoretically yields the exact solution. However, in real computational applications it is not possible to use very large penalty terms, since these lead to ill-conditioned numerical problems, see, e.g., [Lue84]. We will have to bear this issue in mind for the implementation and when analyzing the simulation results, see Chapters 5 and 8.

For extensive analysis and theory of existence and uniqueness, as well as regularity, convergence and stability of solutions for the contact problem of elasticity, we refer to the literature: The mathematical structure of the variational formulation is discussed in detail in, e.g., [RMOC86]. Kikuchi and Oden [KO88] additionally include a mathematical analysis of the FEM for contact problems. Special considerations concerning stability can be found in Klarbring [Kla88]. For dynamic contact problems, existence and uniqueness results are discussed in Martins and Oden [MO87].

Generally, for more details on the theory of elasticity, we refer to the textbooks of Braess [Bra07], Holzapfel [Hol00], Ciarlet [Cia88], and Wriggers [Wri06].

4.2 Modeling the Elastic Behavior of the Mitral Valve

Having in mind Chapter 2 where we described and explained the anatomy of the MV and the surgical procedure of an MVR, and building on top of the above general formulations for modeling the elastic/elastomechanic behavior of biological soft tissues, we can now derive and specify a biomechanical model for the description of the MV and thus for MVR simulation.

Compared to the state-of-the-art in modeling and simulation of the MV and of MVR surgery (see again, Section 3.1), in this work, we do not aim at a further advanced biomechanical model for MV and MVR simulation. We rather aim at developing a *simplified* but *generic* and *flexibly extensible* MV model to represent only the most important functionalities and features of the complex valvular apparatus and of the surgical procedure of MVR annuloplasty. However, through the generic structure and the flexible extensibility, we want to enable future enhancement, e.g., towards more complex mathematical models as referred to above. As such, our mathematical model and the thereon-based surgery simulation are primarily designed to hold for a prototypic implementation of a surgery simulation, which shall serve for research purposes. The simulation therefore is to cover all essential feature types and functionalities, and along with the respective interfaces of its implementation (see Chapter 5) it is to allow for the investigation on its cognition-guided integration into the OR and into cognitive surgery assistance systems.

In the following, in Section 4.2.1, we will detail the derivation and specification of the general elasticity formulations for MV and MVR simulation as published in our works [SES⁺15, SKS⁺15]. In Section 4.2.2, we then discuss current limitations, both model- and integration-wise, and conclude with an outlook to future possible directions.

4.2.1 Derivation and Specification of the General Elasticity Formulation for Modeling and Simulation of the MV and MVR

Based on the above introduced general formulation of the contact BVP of linear elasticity in (eq. 4.100), or its variational formulation in (eq. 4.104), respectively, we can describe the deformation behavior of soft tissue subject to external manipulation. We now want to specify this formulation for the simulation of the MV behavior in order to investigate on the opening and closing functionality of the valvular apparatus and on its stress distribution throughout the entire cardiac cycle, both before and after (virtual) MVR annuloplasty surgery.

To do so, we have to specify the *body region*, the *boundary conditions*, and the *material properties* accordingly, and we will address these items one after the other in the following. We thereby look at the natural MV behavior first, and then go on to the MVR procedure and the post-MVR-surgery behavior.

4.2.1.1 Behavior of the natural MV

As a lesson learned from the state-of-the-art of biomechanical MV and MVR simulation (see again Section 3.1.5), in order to model the behavior of the MV under natural conditions, one has to consider the patient-specific morphology and material properties of MV and ventricle, the blood pressure onto the leaflets and the back-pulling effects of the chordae tendinae.

Additionally, during systole when the valve closes, the event of contact must be handled appropriately.

Patient-specific MV and ventricle morphology. The modeling view of the patient-specific MV and ventricle morphology is pretty straight forward: From 4D TEE ultrasound imaging one can obtain time-resolved 3D segmentations of the MV. In our case, we make use of the segmentations acquired by Engelhardt et al. [ELAM⁺15] in the University Clinic in Heidelberg, Germany. After choosing an appropriate image taken at some time step during diastole (where the MV is open) and possibly performing some smoothing or other enhancement operation on the obtained segmentation, we let these segmented MV representations directly correspond to the body region \mathcal{B} or Ω_R in the above contact elasticity formulation.

Following, extending the details we gave in our work [SES⁺15], we will look at the boundary integrals in (eq. 4.104), and define of a set of adequate boundary conditions (BCs) in order for the simulation to consider the remaining constraints. We remark that – so far – we did not formally distinguish in the notation between *pointwise* and *facetwise* BC constraints. However, in our model (and in the thereon-drawn implementation) we do so, being inspired by the real valvular apparatus and by the MVR procedure, see again, Chapter 2. For all subsequently listed BCs, we will later present the implementation of corresponding manipulation and control interfaces to their respective BC data structures in order to thus allow for patient-specific data input and for surgical MVR knowledge conformity.

Surfacewise Neumann BCs. The blood pressure onto the MV leaflets during the cardiac cycle is represented by means of time-dependent forces that act on the boundary surfaces of the leaflets, i.e., surfacewise/facetwise Neumann BCs. Time-dependence is given in two ways:

On the one hand, the surface forces interpolate through trigonometric functions (of the time variable) the strength of the blood pressure with respect to the upper and lower side of the leaflets according to the mean blood pressure profile by Wiggers, see again Figure 2.4. Specifically, during systole, the high *ventricular pressure* (commonly up to 120 mmHg) acts onto the lower side of the MV leaflets, i.e., onto the side which faces the interior of the left ventricle when the ventricle contracts. This pressure causes the MV to abruptly close and to remain closed (while the AV is open). During diastole, the ventricular pressure is low and the slightly higher *atrial pressure*, which acts onto the upper side of the MV leaflets, as well as the pulling effects of the chordae tendinae (see below) cause the MV to open. For details, see Figure 4.5. We remark that in order for *patient-specifically* time-resolved pressure profiles with respect to the interior MV leaflet side one can intraoperatively make use of so-called pulmonary capillary wedge pressure (PCWP) catheters.

On the other hand, in addition to the time-dependent scaling of these forces, they also change their direction over the course of the cardiac cycle: During systole, the force direction in the 3-dimensional ventricle changes – compare to Figure 4.6 – from towards-the-ventricle-centerline-directed to towards-the-center-of-the-atrium-directed due to the ventricle contraction (such that the valve closes and remains closed). During diastole, it is towards-the-bottom-of-the-valve-directed (such that the valve opens). Please note that these are simplifications, as opposed to reality, where blood flow profiles commonly are not laminar and not directed either,

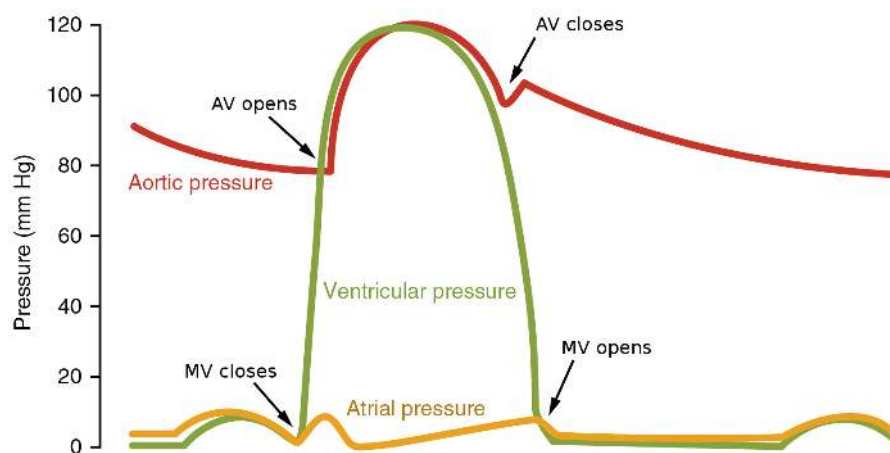


Figure 4.5: Pressure plot over the course of a cardiac cycle, extracted from a Wiggers Diagram, as in Figure 2.4. For modeling the MV behavior we reproduce – on the one hand – the atrial pressure function in what concerns the pressure on the upper MV leaflet sides, and – on the other hand – the ventricular pressure function in what concerns the pressure on the lower MV leaflet sides that face the interior of the left ventricle.

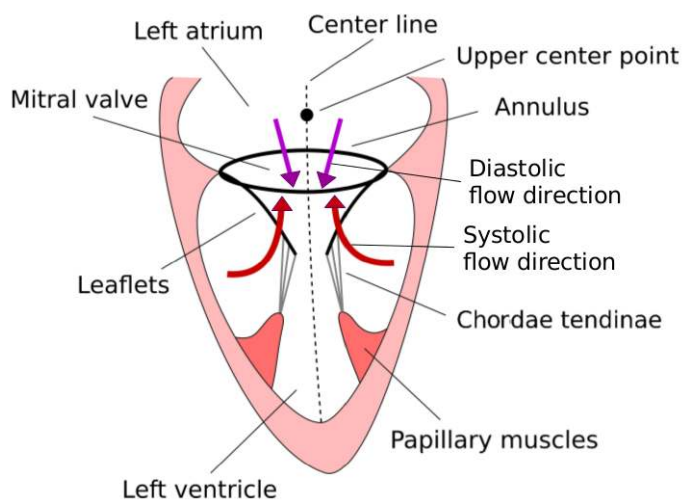


Figure 4.6: Draft of the left ventricle and of the mitral valve apparatus, including the imaginary centerline and blood flow direction tendencies.

but heavily turbulent. Flow tendencies may be obtained by means of 4D MRI measurements in addition to 4D TEE ultrasound imaging, see, e.g., the works of Lau et al. [LDSB10] or of Gao et al [GMQ⁺14].

Summarizing, the effects of the blood are modeled by means of time-dependent facetwise Neumann BCs, which are time-resolved in both their direction and their scaling.

Pointwise Neumann BCs. From the imaging and segmentation results of our clinic partners (see Engelhardt et al. [ELAM⁺15]), we cannot obtain a morphologic representation of the chordae tendinae for our model. However, it is important to properly reproduce their back-pulling effects (see again Figure 4.6 along with the explanations in Chapter 2). In our model, we therefore take them into account by means of pointwise Neumann BCs. These pull down the leaflets when they are pushed up too far due to the blood pressure (i.e., due to the effects of the above surfacewise Neumann BCs).

We represent the primary chordae's attachment points on the leaflets' lower edge according to a mean distribution as suggested by Mansi et al. [MVG⁺12], and we then have the chordae's pulling forces attack at these attachment points and let them pull towards the papillary muscles' positions, see also Figure 4.6. When the MV is wide open, the chordae loosely hang around, however, when the valve is pushed up, they tighten, and gradually – due to their elastic nature [WS12] – cause the back-pulling effect as the distance between the leaflets' lower edge and the papillary muscles becomes too large.

In order to realistically represent this behavior, on the one hand, we scale the strength of the pulling chordae depending on the distance

$$d_{\text{pm}}(\mathbf{c}_i) := \|\mathbf{c}_i - \mathbf{x}_{\text{pm}}\|_2, \quad (4.105)$$

between the papillary muscles \mathbf{x}_{pm} and the chordae attachment points \mathbf{c}_i on the bottom edge of the leaflets. On the other hand, we scale their back-pulling effect by means of a logistic growth function which thus approximately accounts for the chordae's elasticity. Growth starts at the one point in time when the chordae are stretched for the first time in the respective cardiac cycle, and growth ends at the point where they are maximally tightened and thus do not allow the leaflets to be pushed up any further. As the stretch is a function of the distance d_{pm} , we also find the logistic growth as a function of d_{pm} .

This way, we can achieve a smooth (elastic) action of the chordae and realistically represent their restricting effects on the leaflet motion.

Surfacewise Penalty Contact BCs. In order for the simulation to handle the event of contact between the two leaflets, we have to accordingly facilitate and model contact boundaries on the MV leaflet surfaces. For simplification, we assume that either the anterior leaflet gets into contact with the posterior one, or vice versa, but we exclude the possibility of contact between two different leaflet parts of one single leaflet. We denote the surface elements (with respect to the FEM mesh representation, see next Section 4.3.1) of the upper sides of the two leaflets with two different IDs, see Figure 4.7. Due to the anatomy of the valvular apparatus, only the upper sides of the leaflets can potentially get into contact, and the lower sides therefore do not obtain a specific ID in this context.

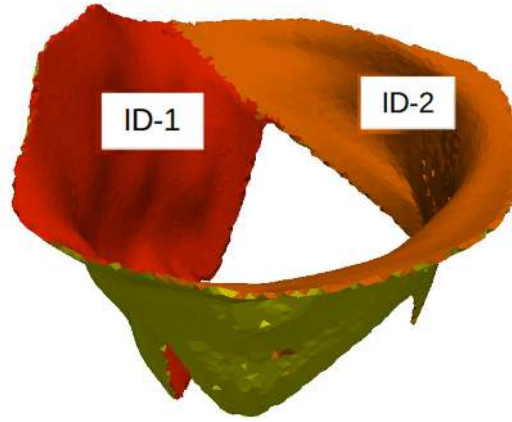


Figure 4.7: Color-coded illustration of the two body parts (leaflets) of the MV (mesh representation), which may get into contact with each other.

The process of the leaflets of getting into touch with each other is then modeled by means of the penalty method as presented in Section 4.1.4. Contact is penalized and the two bodies (i.e., leaflets) are thus prevented from interpenetration. We define contact to be established as from a gap distance d_{gap} of $\varepsilon_{\text{tol}} = 1.5 \text{ mm}$ and lower between the two leaflets, and thus let the penalty term p_C grow towards infinity just before this gap distance shrinks towards 0 mm, according to (eq. 4.97). Moreover, the penalty term is scaled with respect to the velocity of the decrease of this gap distance (counting over the course of three time steps, see next Section 4.3.2). Using small time steps will yield appropriate results, i.e., allow for contact on the one hand and prevent interpenetration on the other hand.

The size of the coaptation zone can then be measured as the size of the contact area, which corresponds to the sum of all surface elements which are less than the above mentioned 1.5 mm away from a surface part of the respectively other leaflet.

Pointwise Dirichlet BCs. In order for the simulation of a cardiac cycle in a natural heart which did not undergo MVR annuloplasty surgery, we would now be done with the description of the effects that influence the behavior of the MV. However, for a complete description in the mathematical sense, i.e., in order to obtain a well-posed problem, we will additionally need a fixed boundary part or a boundary part on which a specific displacement is prescribed. Otherwise, the valve would loosely be pushed around in 3D space due to the acting forces and pressures. Therefore, when not simulating an MVR, one may simply take a set of (preferably equally distributed) points on the natural annulus, which are subjected to fixation, i.e., $\mathbf{u}(\mathbf{x}_i) = \mathbf{0}$, where the \mathbf{x}_i denote the set of these points on the natural annulus. We thus have the Dirichlet boundary part $\Gamma_D = \{\mathbf{x}_i : \mathbf{u}(\mathbf{x}_i) = \mathbf{0}\}$, see Figure 4.8, where this set of points is visualized in blue colors.

Lastly, since we have Dirichlet BCs and Neumann BCs on the surface of the same object (the MV), we have to take care that these do not interfere with each other:

As mentioned above in the theory part in Section 4.1.3.4, the solution of the elasticity

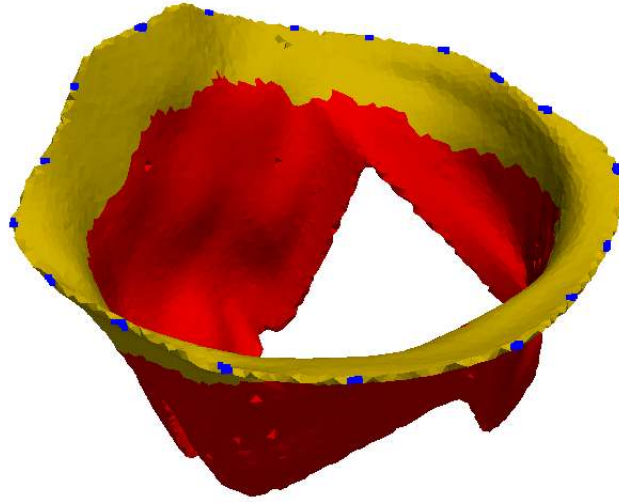


Figure 4.8: Color-coded illustration of the morphology of an MV (mesh representation), where the effective Neumann boundary $\Gamma_N^{\neq 0}$ (red surface) is separated from the Dirichlet boundary Γ_D (blue dots along the annulus) by means of a band (yellow color) which denotes the non-effective zero Neumann boundary Γ_N .

problem may exhibit a certain singular behavior at the boundary interface between Dirichlet boundaries and effective Neumann boundaries due to the potentially unbounded gradient or stress tensor, respectively. In the context of our mitral valve simulation we would thus have to fight this issue pretty often: at each boundary part that is either fixed on the natural annulus or displaced due to prosthesis implantation and at the same time subject to pressure due to blood flow or due to contact.

We therefore introduce by definition a *layer of zero Neumann boundary* Γ_N between the Dirichlet boundary region Γ_D and the effective Neumann boundary region $\Gamma_N^{\neq 0}$ in order to prevent this potential problem. See Figure 4.8 for a visualization of such a preventive layer of zero Neumann boundary around the Dirichlet boundary regions, i.e., around the annulus.

Please note that this issue is also to be taken into account when the occurrence of contact is handled by means of a penalty term, i.e., when contact is represented in a way similar to an additional Neumann boundary condition, see section 4.1.4. In this case, too, it will make sense to introduce a layer of zero Neumann boundary between the Contact boundary and the Dirichlet boundary.

Thus, by definition, we forbid (and accordingly facilitate prevention in a simulation preprocessing step, see Section 6.2.1.3) that Dirichlet BCs can be active in the direct neighbourhood of effective Neumann or Contact BCs.

Mechanical tissue properties of the MV. In order for the proper modeling and simulation of the MV behavior, we need to additionally consider the MV-specific tissue properties.

As already motivated above, in this work, we assume – for reasons of simplicity – to generally deal with a homogeneous, isotropic, objective, linear elastic tissue, represented by the so-called

Saint-Venant-Kirchhoff material model, with the linear stress-strain relation as in (eq. 4.60). However, this is clearly different from real MV tissue, where nonlinear and anisotropic effects do occur, and which consists of several inhomogeneous sub-components all of which have different material properties. For instance, it is today common believe that the MV leaflets must be considered as three-layered laminated structures with different mechanical properties each [PS09]. Also, their behavior is highly anisotropic and nonlinear for certain stages during the cardiac cycle, such that models relying on transversely isotropic finite elasticity are agreed upon to be most appropriate for MV simulation [MPW⁺16].

We are hence well-aware of the fact that simulation accuracy and realistic simulation behavior will suffer from our simplifications [SES⁺15]. Particularly, as mentioned above, ghost forces that blow up the body are to be expected when deformations become significantly large.

Nevertheless, our focus lies on the investigation on the integration of a biomechanical MVR simulation into cognitive surgery assistance, so we rather aim at setting up a basic but generic and flexibly extensible model (in which – if required – more advanced elasticity formulations may still be included later).

This being said, we have learned from the state-of-the-art of biomechanical MV and MVR simulation in Section 3.1.5, that in-vivo examination as opposed to ex-vivo experiments has revealed the true characteristics of these materials. Also, several research groups found that tissue properties notably change between healthy and pathological conditions, as well as between different pathological states (e.g., functional mitral regurgitation vs. myxomatous disease). In contrast, the inter-patient variations of mechanical properties within specific clinical scenarios seems not to be as relevant as the variations in morphological valve features [PSSH10].

This lead us to the decision to *classify* ranges of tissue properties and of tissue properties combinations according to these specific clinical scenarios. Then employing *mean values* for each of the resulting classes in the respective MV or MVR simulation seems to be very reasonable for our prototyping purposes.

Especially, when dealing with MVs that show the particular type of pathologic behavior which requires subsequent treatment by means of MVR annuloplasty surgery, we find suggested tissue properties distributions, e.g., in the works of Mansi et al. [MVG⁺12] or Pouch et al. [PXY⁺12]:

- Tissue density: $\rho = 1.04 \frac{\text{g}}{\text{mL}}$.
- Poisson's Ratio: $\nu = 0.488$ for both anterior leaflet (AL) and posterior leaflet (PL).
- Young's Modulus: $E = 6.23/2.35$ MPa (AL) and $E = 2.08/1.89$ MPa (PL) for the leaflet fibers in circumferential/radial direction.
- Shear modulus: $\mu = 1.37$ MPa (AL) and $\mu = 0.69$ MPa (PL).

As mentioned earlier in the theory section, in our MV model and simulation, we assume the tissue to be *homogeneous, objective, isotropic, and linear-elastic*. We employ the Saint-Venant-Kirchhoff material model, with the linear stress-strain relation as in (eq. 4.60), and therefore transform the above stated tissue parameters into the two Lamé constants, which yields

- Lamé's first parameter λ in a range of 28 466 Pa and 68 640 Pa, and
- Lamé's second parameter or shear modulus μ in a range between 700 Pa and 1 400 Pa.

Experimentally, we found that an average parameter combination of $\lambda = 56\,933\text{ Pa}$ and $\mu = 1\,400\text{ Pa}$ yields realistic results for the major part of simulated patient MVs. Minor deviations require for manual calibration, e.g., through comparison with imaging results.

Please note that simulation results which do not properly represent experimental results do not necessarily derive from wrong material parameters, but may also be due to an unsuitable elasticity model, see again Section 4.1.3. Specifically, in the case of this work, when a linear elastic material model is used, the tissue may blow up unnaturally or exhibit an overly *stiff* behavior when displacements are too big and deformations become too large. This may then be due to *ghost forces* or due to *numerical locking effects*, which likely occur for thin and almost incompressible soft tissues (such as the MV leaflets with an average thickness of 1.2mm [MVG⁺12] and a Poisson Ratio of almost 0.5 due to their strongly hydrated nature). Other material models, e.g., the corotation-based formulation, may then be more suitable.

We outlined this above in Section 4.1.3, and refer to literature for more information [Bra07], [Hol00], [SSS⁺13], [Mez08], [Suw14]. We emphasize again, that we aim at the derivation of a simple but generic and flexibly extensible model for MV/MVR simulation which is to hold as a research prototype in order for investigating on its integration into a cognitive surgery assistance system.

Along with the material parameters, in order to reproduce to some extent the MV's viscoelastic behavior, we apply *Rayleigh Damping* as motivated in Section 4.1.3. This also increases system robustness under strong collisions, i.e., when the leaflets are pushed towards each other during the valve closing process. Mansi et al. [MVG⁺12] suggest a linear combination of $\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K}$ as in (eq. 4.56) with $\alpha = 0.1s^{-1}$ and $\beta = 0.1s$ as a good compromise between system damping and robustness according to non-reported experiments. In our own experiments, we figured the best-suitable setting for the Rayleigh Damping to be obtained with $\alpha = 0.3s^{-1}$ and $\beta = 0.3s$. As a point of reference, this combination approximately damps an oscillation, which is induced by an instant pressure of 40 mmHg onto the leaflets, in a way such that after 20 seconds it is completely damped, i.e., gone.

It is to mention at this point, that due to the thin leaflets and to the strong (blood pressure-induced) forces acting on the leaflets, we generally neglect *gravity* in our simulation. Also, as suggested by Votta et al. [VCV⁺08], we artificially scale the *simulation time* such that the simulated valve closure is ten times *slower* than what is observed in reality (from 70 to 150 ms), which is needed in order to handle the strong (and discontinuous) Neumann and Contact forces.

4.2.1.2 Behavior of the MV during and after MVR Annuloplasty Surgery

Following the thus facilitated delineation of the natural behavior of a healthy MV, we can now focus on the surgical procedure of an MVR, and describe how we want to facilitate the virtual annuloplasty ring implantation and its post-surgery physiological effects by providing adequate additional BCs for our model and simulation. Again, we extend the details given in our work [SES⁺15].

Pointwise Dirichlet BCs. First, we consider the implantation procedure of the annuloplasty ring prosthesis. As described in Section 2.2, an almost inelastic artificial ring prosthesis is sutured and fixed onto the natural mitral annulus, see again Figure 2.7.

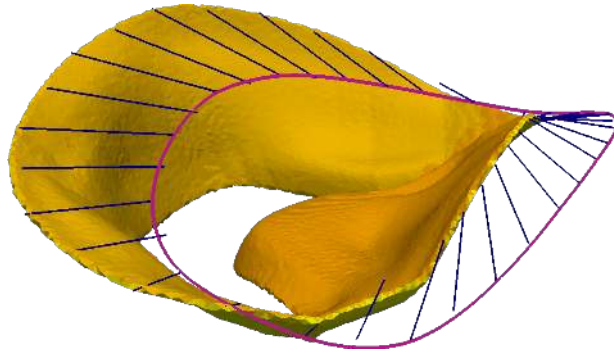


Figure 4.9: Illustration of an MV (mesh representation) and of the pointwise Dirichlet BCs along with their respective displacement vectors (blue) to model the implantation procedure of an annuloplasty ring prosthesis (violet).

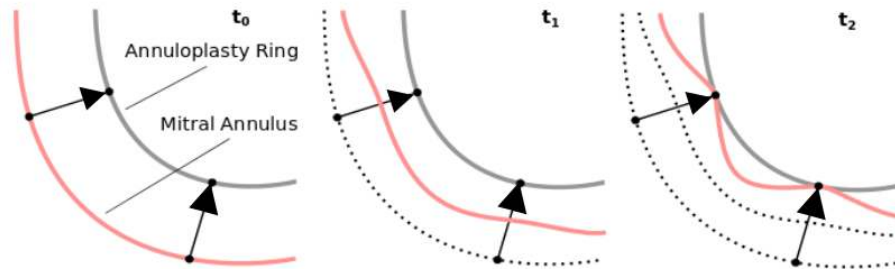


Figure 4.10: Illustration of the time-dependent pointwise Dirichlet BCs to represent the implantation procedure of the annuloplasty ring prosthesis, where a discrete set of suture points on the natural mitral annulus is displaced (by means of the suture) onto the artificial annuloplasty ring prosthesis.

We thus have to model the pulling forces of the suture which acts on a discrete set of anatomically distinguished suture points on the natural mitral annulus. The respective counterparts of these suture points are marked on the artificial annuloplasty ring, such that for each suture point on the natural annulus there is a distinguished point on the artificial annuloplasty ring and vice versa; see Figure 4.9 and compare to Figure 2.7. We model the pulling effect of the suture on the suture points by means of time-dependent pointwise Dirichlet BCs: Each suture point on the natural annulus is displaced along a straight line (the *displacement vector*) to its corresponding counterpart point on the artificial ring, such that each point reaches its destination with constant velocity after 2 seconds at t_2 ; see Figure 4.10.

Please note that these newly defined pointwise Dirichlet BCs do replace the before-mentioned pointwise fixation Dirichlet BCs, which – in the natural case – were to fix the natural annulus at its natural position.

During the entire surgical process, the heart is bypassed, and there are hence no other external forces or momentums that would affect the cardiac tissue. Still, due to the strong pulling forces of the suture which cause the natural annulus to be drawn to the imposed setting

on the artificial annuloplasty ring, in the model and simulation there may occur temporary instabilities at the respective points where these forces are represented by means of pointwise Dirichlet BCs. Therefore, in order to increase stability, we extend the above pointwise Dirichlet BCs to the surrounding boundary element surfaces (with respect to the MV mesh in the Finite Element context below) and thus actually obtain **surfacewise Dirichlet BCs** (which, however, act on rather small surface parts).

Also due to the rapid pulling forces acting during the implantation procedure a strong acceleration affects not only the annulus, but also the leaflets. When the suture destination on the annuloplasty ring prosthesis is reached, there is a sudden stop which causes the leaflets to swing on and to enter an oscillation process. This oscillation is gradually decaying and we therefore accordingly let the valve (or rather its morphologic representation) swing off for five seconds, such that an almost vibration-free state of balance is regained. The decay of the oscillation is enforced through *Rayleigh Damping*, as mentioned above and introduced in Section 4.1.3.3.

Other BCs as above. After the virtual ring implantation, the surgeons are now able to finally look at their virtual post-operative results. Thinking further, they are, of course, interested to know about the MV's post-operative opening and closing functionality (during a virtual post-operative cardiac cycle). This, however, would so far be completely based on their surgical experience and on their ability to imagine the potential tissue behavior in 3D from only looking at the morphology.

In our computer simulation, at this point, we therefore want to simulate a post-implantation cardiac cycle in order to enable the surgeon to virtually assess the post-operative opening and closing behavior of the valvular apparatus. Therefore, we again apply the above conditions that we used to model and simulate the MV's natural behavior during the natural cardiac cycle (Section 4.2.1.1). These hold for situations both before and after MVR surgery.

Concluding, in this section, we have specified the above general contact elasticity problem formulation for the particular case of modeling and simulating the MV behavior before during and after MVR annuloplasty surgery.

We have found that the general formulation of the contact BVP of linear elasticity as in (eq. 4.100) along with the respective function spaces as in (4.101) to (4.103)

- is put into context with the patient-specific MV morphology, to describe the shape and size of the soft body,
- that it requires for the definition of the boundary conditions (BCs), to describe the behavior under natural conditions during the cardiac cycle and/or under surgical manipulation, and
- that the material parameters need to be set appropriately, in order to properly describe the MV's soft elastic behavior.

The same specifications hold for the respective variational formulation of the contact elasticity problem as in (eq. 4.104). See Figure 4.11 for a summarizing illustration.

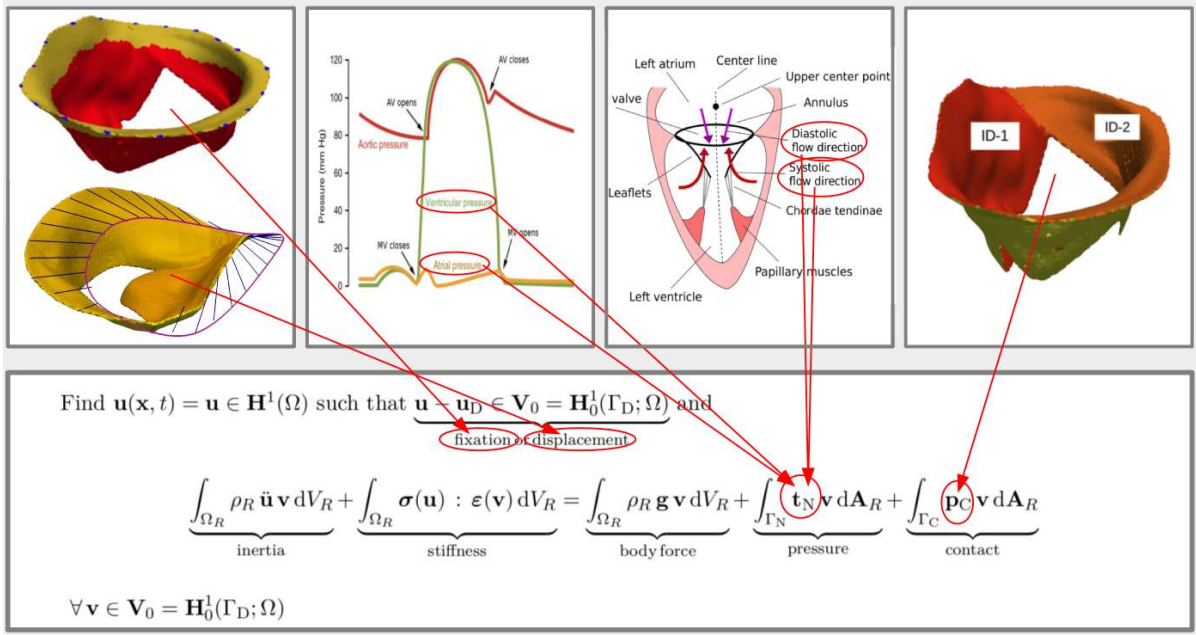


Figure 4.11: Summarizing illustration of the specifications of the general contact elasticity problem formulation as in (eq. 4.104) that were made for describing the MV behavior and the procedure of MVR annuloplasty surgery.

With a view to the subsequent explanations in the following chapters, we will not differentiate anymore between pointwise and surfacewise boundary conditions in what concerns the integrals in (eq. 4.104) for reasons of better readability. Mathematically, they will be handled equally. Yet, of course, in the implementation chapter, we will again give according remarks on the respective implementation differences.

Lastly, for a detailed analysis of the existence and uniqueness of solutions of the above specified contact elasticity problem, we refer to the before-mentioned literature. In particular, we again emphasize the work of Martins and Oden [MO87], who provide a general discussion of existence and uniqueness for dynamic contact problems, and the work of Kikuchi and Oden [KO88], who have conducted a mathematical analysis of the FEM for contact elasticity problems. Also, we again point out the respective, above-stated assumptions and requirements (see Sections 4.1.3.4 and 4.1.4), which either are naturally satisfied, or are adequately enforced and adapted, e.g., in a modeling and simulation preprocessing step, see Chapter 6.

4.2.2 MV Modeling and Simulation: Current Limitations

Before going on with the numerical solution of the above derived specific formulation of the contact elasticity problem for the MV and for MVR, we want to point out the most important limitations of our model and their respective implications on the simulation results.

- **Representation of the chordae tendinae:** As mentioned above, the imaging techniques used by our clinic partners were not able to reasonably depict the chordae tendinae [ELAM⁺15], which is why in our model – as a substitute – we chose to assume the primary chordae to be pseudo-elastic and equally distributed along the leaflets’ lower edge, pointing towards an average location of the papillary muscles. This is, at best, heuristically motivated, and we thus neither make a difference between primary and secondary chordae, nor do we consider patient-individual distributions, dimensions or mechanical properties, both of which were reported to have an important impact on the valvular kinematics [MPW⁺16, CYMN04, PHS09].
- **Linear elasticity model:** For reasons of simplicity and of higher computational efficiency, we employ a linear elastic model for the simulation of partly large deformations of the MV leaflets. In addition to linear elasticity, we assume the tissue to be homogeneous, objective and isotropic, and we make use of the Saint-Venant-Kirchhoff material model, even though several works have shown that this is not fully suitable for MV tissue [HGO00, PS09].
- **Uniform distribution of material properties:** In our work, we assume a uniform leaflet thickness and a homogeneous distribution of material properties, for both leaflets and the annulus. We do not make a difference between diseased and healthy MVs either. Opposed to this, Pouch et al. [PXY⁺12] have shown that material characteristics may vary regionally even in a single valve, and Mansi et al. [MVG⁺12] were able to capture the leaflet thickness distribution, which was shown to strongly affect the leaflets’ stress patterns. Moreover, the leaflets have been shown to be constituted as three-layered laminated structures with different mechanical properties in each layer [PS09].
- **No Fluid-Structure Interaction (FSI):** In contrast to many pioneering recent works on MV tissue simulation, we assume pressure profile to account for the effects of the blood flow, which of course does not cover the whole spectrum of cardiac flow-related phenomena, compare to [VLS⁺13], [Mar14] or [MPW⁺16].
- **Contact model:** Instead of the above motivated contact model, which is based on the penalty formulation, one might implement contact as an optimization problem (Signorini’s formulation), too. This is supposed to yield more stable results even for larger time steps, and hence may even be more efficient with respect to the computation time [Wri06].
- **No Uncertainty Quantification:** We are aware of the fact that we built up our model (and the thereon-based simulation prototype) on partly suboptimal or inappropriate assumptions, which may cause variations between real experimental results and simulation results. Instead of neglecting this circumstance and not accounting for the issue that the real simulation parameters may actually vary, the application of methods for Uncertainty Quantification might have the potential to partly cover the thus caused distortions, and to thus yield more reliable simulation results [LMK10].

As mentioned before, compared to the state-of-the-art (Section 3) we do not aim for perfection of our MV and MVR simulation. Rather, with this simplified model and the thereon-based

simulation, we want to facilitate our ongoing research which focuses the integration of a numerical soft tissue simulation into a cognitive surgery assistance system for cardiac surgery.

In the following section, we therefore look at the numerical solution of the above specified contact elasticity problem for the MV and for MVR by means of the Finite Element Method (FEM). Thereafter, we will be able to deal with the aforesaid cognitive integration of our FEM simulation into surgery assistance.

4.3 Numerical Solution of the Contact Elasticity Problem for the Mitral Valve

Unfortunately, the above introduced space and time continuous contact elasticity problem (eq. 4.100) (or its variational formulation (eq. 4.104), respectively) cannot be solved analytically (with only a few simple exceptions). Therefore, we want to find a numerical approximation of the solution of the variational formulation (eq. 4.104).

In this section, extending the description in our work [SES⁺15], we will describe the procedure to numerically solve the (space and time continuous) variational formulation of the contact elasticity problem for the MV by means of the so-called *method of lines*. This means that first the spatial derivatives are discretized while the time derivative is left continuous, see Section 4.3.1. We thereby follow the so-called conforming Galerkin approach and apply the Finite Element Method (FEM). This leads to a system of ordinary differential equations (ODEs) in time that can be solved using time discretization algorithms, see Section 4.3.2. We outline our approach for the linear elastic case only. However, it generalizes very well to the nonlinear problem.

4.3.1 Discretization in Space – The Finite Element Method (FEM)

Partial differential equations (PDEs) are generally very difficult or even impossible to solve analytically. The above introduced contact elasticity problem (eq. 4.100) which describes the MV behavior, respectively after MVR surgery, cannot be solved analytically either, and we want to find a numerical approximation of the solution.

Simply switching over to the variational formulation (eq. 4.104) at first glance does not seem to give any benefits in this respect, and we cannot expect to be able to solve the continuous (infinite-dimensional) problem as in (eq. 4.104) in $\mathbf{H}^1(\Omega)$ either. Therefore, for the numerical solution, we follow an approach known as the *conforming Galerkin method*: We replace the infinite-dimensional space $\mathbf{H}^1(\Omega)$ by some finite-dimensional subspace $\mathbf{V}_h \subset \mathbf{H}^1(\Omega)$ to obtain a finite-dimensional problem, which we will solve using the Finite Element Method (FEM).

The core idea of the FEM then is to determine the exact solution for the problem in the discrete (finite-dimensional) subspace $\mathbf{V}_h \subset \mathbf{H}^1(\Omega)$. In order to achieve that goal, one requires the solution to hold for the variational formulation and only with respect to specific test functions.

Therefore, first, the variational form is discretized by means of the conforming Galerkin method into a (time dependent) system of linear ordinary differential equations (ODEs). Second, we define the above-mentioned discrete solution space and its specific properties (such as nodal basis, small supports of the basis functions, etc.) according to the idea of the FEM.

4.3.1.1 The Galerkin Method

In the first instance, we assume the above-mentioned discrete (finite-dimensional) approximation space \mathbf{V}_h to already be defined along with its basis functions and all geometrical and topological properties. Thereon-based, we start to develop the characteristics of the discrete solution, which is also known as the *Galerkin solution*, \mathbf{u}_h , and will get back to the properties of the solution space \mathbf{V}_h (along with its basis, etc.) afterwards.

Starting from (eq. 4.104), we hence replace $\mathbf{H}^1(\Omega)$ by the *finite-dimensional* subspace $\mathbf{V}_h \subset \mathbf{H}^1(\Omega)$, and obtain the finite-dimensional problem:

$$\begin{aligned}
 & \text{Find } \mathbf{u}_h \in \mathbf{V}_h \text{ such that } \mathbf{u}_h - \mathbf{u}_{h,D} \in \mathbf{V}_{h,0} = \mathbf{V}_h \cap \mathbf{H}_0^1(\Gamma_D; \Omega) \text{ and} \\
 & \underbrace{\int_{\Omega_R} \rho_R \ddot{\mathbf{u}}_h \mathbf{v}_h \, dV_R}_{\text{inertia}} + \underbrace{\int_{\Omega_R} \boldsymbol{\sigma}(\mathbf{u}_h) : \boldsymbol{\varepsilon}(\mathbf{v}_h) \, dV_R}_{\text{stiffness}} \\
 & = \underbrace{\int_{\Omega_R} \rho_R \mathbf{g} \mathbf{v}_h \, dV_R}_{\text{body force}} + \underbrace{\int_{\Gamma_N} \mathbf{t}_N \mathbf{v}_h \, dA_R}_{\text{pressure}} + \underbrace{\int_{\Gamma_C} \mathbf{p}_C \mathbf{v}_h \, dA_R}_{\text{contact}}, \quad (4.106) \\
 & \forall \mathbf{v}_h \in \mathbf{V}_{h,0} = \mathbf{V}_h \cap \mathbf{H}_0^1(\Gamma_D; \Omega).
 \end{aligned}$$

The discretization parameter h indicates that we want to achieve convergence of the *discrete* solution \mathbf{u}_h against the solution of (eq. 4.104) for $h \rightarrow 0$.

When working in 3D, let $3N = \dim \mathbf{V}_h < \infty$ be the dimension of \mathbf{V}_h , and choose a basis $\{\phi_i : i = 1, \dots, 3N\}$ of \mathbf{V}_h . Then we may represent the (semi-)discrete solution $\mathbf{u}_h = \mathbf{u}_h(\mathbf{x}, t)$ with respect to this basis as:

$$\mathbf{u}_h(\mathbf{x}, t) = \sum_{i=1}^{3N} u_i(t) \phi_i(\mathbf{x}), \quad (4.107)$$

where the vector $\mathbf{u}_h(t) = (u_1(t), \dots, u_{3N}(t))^T \in \mathbb{R}^{3N}$ contains the uniquely determined (time-dependent) coefficients $u_i(t)$ of the linear combination. Please note that we now use vector style notation, i.e., \mathbf{u} instead of \mathbf{u} , since we consider the *discrete, finite-dimensional Galerkin solution* instead of the continuous, infinite-dimensional functional \mathbf{u} , which we were using in (eq. 4.104).

Inserting this basis representation for $\mathbf{u}_h(\mathbf{x}, t)$ and for $\mathbf{v}_h(\mathbf{x}, t)$ into (eq. 4.106), and canceling out the coefficients $v_j(t)$ of $\mathbf{v}_h(\mathbf{x}, t)$ (which we are allowed to do due to the linearity of the integration), we obtain

$$\begin{aligned}
 & \sum_{i=1}^{3N} \partial_t^2 u_i(t) \int_{\Omega_R} \rho_R \phi_i(\mathbf{x}) \phi_j(\mathbf{x}) \, dV_R + \sum_{i=1}^{3N} u_i(t) \int_{\Omega_R} \boldsymbol{\sigma}(\phi_i(\mathbf{x})) : \boldsymbol{\varepsilon}(\phi_j(\mathbf{x})) \, dV_R \\
 & = \int_{\Omega_R} \rho_R \mathbf{g}(t) \phi_j(\mathbf{x}) \, dV_R + \int_{\Gamma_N} \mathbf{t}_N(t) \phi_j(\mathbf{x}) \, dA_R + \int_{\Gamma_C} \mathbf{p}_C(t) \phi_j(\mathbf{x}) \, dA_R. \quad (4.108)
 \end{aligned}$$

for $j = 1, \dots, 3N$. Please note that in the finite-dimensional space \mathbf{V}_h , it hence is sufficient to only test the variational formulation equation against the (finite) set of basis functions.

Next, we introduce the matrices \mathbf{K} (stiffness matrix) and \mathbf{M} (mass matrix), and the right-hand side vector $\mathbf{f}(t)$ as

$$\begin{aligned} \mathbf{K}_{ji} = k(\phi_j, \phi_i) &:= \int_{\Omega_R} \boldsymbol{\sigma}(\phi_i) : \boldsymbol{\varepsilon}(\phi_j) dV_R \\ &\stackrel{(4.60)}{=} \int_{\Omega_R} (\lambda (\operatorname{tr} \boldsymbol{\varepsilon}(\phi_i)) \mathbf{I} + 2\mu \boldsymbol{\varepsilon}(\phi_i)) : \boldsymbol{\varepsilon}(\phi_j) dV_R \\ &= \int_{\Omega_R} \lambda \operatorname{div} \phi_i \operatorname{div} \phi_j + 2\mu \boldsymbol{\varepsilon}(\phi_i) : \boldsymbol{\varepsilon}(\phi_j) dV_R, \end{aligned} \quad (4.109)$$

$$\mathbf{M}_{ji} = m(\phi_j, \phi_i) := \int_{\Omega_R} \rho \phi_i \phi_j dV_R, \quad (4.110)$$

$$\mathbf{f}_j(t) = f(\phi_j)(t) := \int_{\Omega_R} \rho \mathbf{g}(t) \phi_j dV_R + \int_{\Gamma_N} \mathbf{t}_N(t) \phi_j dA_R + \int_{\Gamma_C} \mathbf{p}_C(t) \phi_j dA_R \quad (4.111)$$

for $i, j = 1, \dots, 3N$. We have omitted the argument \mathbf{x} in $\phi_{i,j} = \phi_{i,j}(\mathbf{x})$ for reasons of better readability. Now, we can rewrite the above integral equation as a vector-valued equation

$$\mathbf{M} \ddot{\mathbf{u}}_h(t) + \mathbf{K} \mathbf{u}_h(t) = \mathbf{f}(t), \quad (4.112)$$

or, respectively,

$$\mathbf{M} \ddot{\mathbf{u}}_h(t) + \mathbf{D} \dot{\mathbf{u}}_h(t) + \mathbf{K} \mathbf{u}_h(t) = \mathbf{f}(t), \quad (4.113)$$

with the additional damping matrix $\mathbf{D} = \alpha \mathbf{K} + \beta \mathbf{M}$ according to the above-introduced approach of Rayleigh: We remark that we will make use of the Rayleigh constants α and β in our simulation application scenarios in order to calibrate and control energy dissipation, swinging behavior, and the overall simulation stability.

Via the discretization approach as suggested by the Galerkin method, solving the (semi-) discrete formulation (eq. 4.113) for the displacement vector \mathbf{u}_h is hence equivalent to solving the variational formulation (eq. 4.106) of the contact elasticity problem.

We point out that both pressure and contact boundary conditions are considered by means of the boundary integrals in (eq. 4.111), and thus reflected in the formulation through the right-hand side vector $\mathbf{f}(t)$. However, Dirichlet (i.e., prescribed displacement) boundary conditions are not yet taken into account, and will be taken care of by means of a specific procedure, see Section 4.3.2.3.

4.3.1.2 Finite Elements

We have so far shown how – following the conforming Galerkin approach – from the variational formulation in (eq. 4.106) a linear system of ODEs in time (eq. 4.113) can be derived, after an approximation space \mathbf{V}_h with a basis $\{\phi_i\}$ has been chosen. It is clear that different choices of spaces will lead to different linear systems, and the choice of the basis will strongly influence the numerical properties of the arising system matrices.

In this paragraph, we will address the question of how the discrete space \mathbf{V}_h and its basis can be defined, and thereby follow the idea of the Finite Element Method (FEM), which can be interpreted as a Galerkin method for discrete spaces with specific properties. There are several factors that play a role in the choice of \mathbf{V}_h . First, we require that $\mathbf{V}_h \subset \mathbf{H}_0^1(\Omega)$ holds. Second,

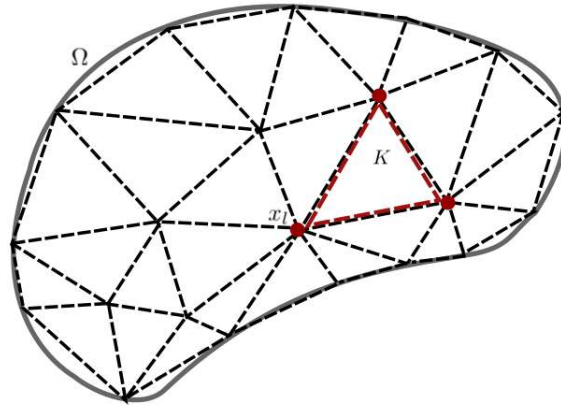


Figure 4.12: Triangulation of a 2D image representation of a liver. The body Ω is discretized by means of triangular cells K , that are defined through a set of N nodes x_l , with $l \in \{1, \dots, N\}$.

the larger we choose the space \mathbf{V}_h , the better we can hope to approximate the real solution on the one hand. However, on the other hand, the *dimension* of the linear system which we have to solve will become larger, too. We will explicitly address this compromise between approximation quality and convergence behavior below, when looking at the properties of the thus resulting FEM-based discrete solution.

In order to be able to store and solve large equation systems it is advantageous if the system matrices are *sparse*, i.e., almost all matrix entries are zero. As a specific feature, the FEM provides such sparse matrices by defining the space in terms of basis functions that are *piecewise continuous polynomials* with a very *localized support*. This implies that most pairs of basis functions (ϕ_i, ϕ_j) will satisfy $k(\phi_j, \phi_i) = 0$ and $m(\phi_j, \phi_i) = 0$. Only those pairs whose supports are such close that they overlap, will yield non-zero entries in the matrices.

The supports of the basis functions are typically defined through a *mesh*, which is a partition of Ω into *cells* of a simple shape, such as triangles or quadrilaterals in 2D, and tetrahedra or hexahedra in 3D, and generally polyhedral elements, see Figure 4.12. The process of the *mesh generation* is known as *triangulation*. In this context, the discretization parameter h refers to the maximum diameter of the mesh cells, and $h \rightarrow 0$ means *refinement* of the mesh. Also, the boundary of Ω is required to be sufficiently smoothed, which is formalized through the assumption that Ω is a Lipschitz domain. In real applications in the medical context this is usually satisfied.

The formal definition of a Finite Element, which Ciarlet created in 1988 [Cia88], is then as follows: A Finite Element is a triplet (K, P, Σ) , where

- $K \subset \mathbb{R}^d$ is a polyhedral cell of geometrical dimension d ,
- $P(K)$ is a space of functions defined on K ,
- $\Sigma = \{\sigma_1, \dots, \sigma_N\}$ is a basis of $\mathcal{L}(P, \mathbb{R})$; the σ_i are called *degrees of freedom* (DoFs).

Usually, a *basis* of local shape functions ϕ_i^K for $P(K)$ is chosen such that $\sigma_i(\phi_i) = \delta_{ij}$, which is called the *nodal basis*. Through selection of the DoFs the basis is thus uniquely determined.

Generally, one can construct miscellaneous types of elements with different cells, shape functions and DoFs. Here, we will restrict ourselves to the most common type, which is known as the *Lagrange* class of elements. We choose the polynomial degree p of the functions in P , and a set of N nodal points $x_l \in K$ with $\sigma_l(w) = w(x_l)$ for $w \in P$. The local shape functions ϕ_i^K which satisfy $\phi_i(x_l) = \delta_{il}$ will then be exactly the Lagrange basis polynomials for the nodal points x_l . For more information, please see [Cia88]. As a direct consequence of the δ -property, the coefficients $u_i(t)$ coincide with the displacements of the element nodes when using nodal basis functions.

After having defined one element for each cell in the mesh, the global basis functions ϕ_i are created by ‘glueing together’ the local shape functions on neighboring elements. It can be shown that, in order for the global basis functions and hence also for the solution and test functions to lie in $\mathbf{H}^1(\Omega)$, they have to be globally continuous (C^0 -continuous), see, e.g., Braess [Bra07]. The most common way to enforce the continuity for Lagrange elements when the same polynomial order is used on all elements, is to identify the DoFs that are shared by neighboring elements via their numbering. Conformity with the concept of the partition of unity is thus guaranteed, too. Working in 3D and using (linear) Lagrange FEs, a mesh with N nodes will thus yield a linear system of dimension $3N$. For more details, we refer to Braess [Bra07] or Ciarlet [Cia88].

We conclude with the remark that it is only through this so-called conformal FE space that statements on uniqueness and existence of solutions (i.e., on the well-posedness of the problem) can directly be transferred from the continuous to the discrete problem.

When constructing the (discrete) finite element space \mathbf{V}_h , one faces the basic problem associated with every discretization technique: How can the best approximation \mathbf{u}_h to the correct solution \mathbf{u} be found with a given amount of complexity in terms of the DoFs of the discrete system – or in terms of the coefficients of the DoFs of the discrete system, respectively. Thus looking at the relation between the choice of the discrete FE space and the resulting specific properties of the FEM-based discrete solution, namely its approximation quality and convergence behavior, one of the reasons for the success of the FEM is that there are powerful and elegant tools available for *a-priori error approximation*.

At the core, *Céa’s Lemma* states that the finite element approximation \mathbf{u}_h is the near best fit to the solution \mathbf{u} in the norm associated with \mathbf{H}^1 [Bra07]. Geometrically speaking, the discrete solution \mathbf{u}_h is the *orthogonal projection* of \mathbf{u} into \mathbf{V}_h with respect to the given inner product. Consequently, the construction of the subspace \mathbf{V}_h is crucial for the accuracy of the FE method.

The *convergence* of FE methods is typically studied in suitable mesh-dependent norms. There exist several a-priori error estimates for different norms and different finite element types. For finite element spaces in 3D that are spanned by piecewise continuous polynomials of order p on tetrahedral and hexahedral elements, it can be shown that the convergence rate is of order $p + 1$, if the real solution is *sufficiently smooth*. However, in many applications in the medical assistance context the latter is commonly not the case. Generally, for elasticity problems one typically uses first or second order polynomials. In the medical simulation context, com-

binning these with tetrahedral elements has proven particularly suitable. When dealing with thin, almost incompressible tissues, *quadratic, tetrahedral elements* (so-called Tet10-elements) have thus shown to often mitigate the typical numerical volume locking problems [CK92]. The respective nodal basis functions for linear and quadratic tetrahedral elements, which we use for our MVR soft tissue simulation, can be found in the standard literature, e.g., Bathe [Bat96].

Concluding, we found that using the Galerkin approach for space discretization, we obtain – for a 3D triangulation with N nodes – a linear system of $3N$ second order ODEs in time. This (semi-) discrete formulation (eq. 4.113) has to be solved for the displacement vector $\mathbf{u}_h(t) \in \mathbb{R}^{3N}$. As the test functions are polynomials, the respective surface and volume integrals in (eq. 4.109) to (eq. 4.111) can accurately and efficiently be evaluated using numerical integration techniques such as Gaussian quadrature, see Section 5.1. For an overview on this field, we refer to [ZT00].

From the relations (eq. 4.109), (eq. 4.110), (eq. 4.111), and (eq. 4.113), it is apparent how the global matrices \mathbf{M} , \mathbf{D} and \mathbf{K} , and the global right-hand side vector $\mathbf{f}(t)$ can be constructed. First the integrals are evaluated on a per-element basis. Due to the small support of the basis functions, only few integrals (e.g., 12 for linear tetrahedra) have to be computed for each row. The elemental building blocks are then added into the global data structures (which is known as *matrix or vector assembly*) via their lexicographical numbering. Hence, the computation of the matrix and vector entries can directly be found in the assembly parts of our implementation, see Section 5.2.

Resulting, we find that – based on the relations (eq. 4.109) and (eq. 4.110) – the global matrices \mathbf{M} , \mathbf{D} and \mathbf{K} are *symmetric, positive definite, sparse* matrices. Their sparsity pattern particularly depends on the mesh topology and is thus irregular for unstructured grids.

According to the method of lines, we now have to apply some time discretization algorithm in order to numerically solve the obtained system of second order ODEs in time; see the following Section 4.3.2.

4.3.2 Discretization in Time – RK4 and Newmark Time Integration

In this section, we will derive the full discretization, i.e., we discretize the second order ODE (eq. 4.113) in time to derive a linear system of equations. Firstly, we divide the time interval $T > 0$ (which in our case usually covers the time of the simulated MVR procedure and one subsequent cardiac cycle) into an equidistant grid of points. Choosing to have $M \in \mathbb{N}$ time steps, we let $\Delta t = \frac{T}{M}$ be the resulting time step size. Then the grid is given by $t^n := n \cdot \Delta t$, with $n = 0, 1, \dots, M$.

Generally, time integration methods can be categorized into explicit and implicit methods. The choice of the method to determine the motion of the body very much depends upon the problem at hand. Explicit methods enforce the equilibrium in (eq. 4.113) only at the beginning of the time step at the current time t , whereas implicit algorithms enforce it at the end of the time step at time $t + \Delta t$.

Explicit methods are easy to implement, and very computationally efficient, as their computation only requires matrix-vector operations. It is important to note that this holds even for nonlinear elasticity, since all nonlinearities only enter the equations on the right-hand side. Fur-

thermore, especially when the mass matrix is approximated by a lumped mass matrix (which is diagonal) explicit algorithms are extremely efficient. In contrast, implicit methods require a system of (possibly nonlinear) equations to be solved at each time step [Ran08], and are hence a lot more expensive, since they have to be combined with, e.g., Newton-Raphson procedures in order to solve for nonlinear elasticity.

However, explicit methods are only *conditionally stable*. Especially for stiff systems, they need very small time steps to remain stable, and the time step size is governed by the *Courant criterion* [Wri06]. Consequently, they are often a poor choice for soft tissue simulations, particularly for almost incompressible material models [SM03]. Therefore, for real-time deformable model problems, implicit methods have emerged as the dominant means for time discretization [BW98]. As opposed to explicit methods, implicit algorithms can be constructed in a way such that they are *unconditionally stable*, and hence can be applied with a far bigger time step size than explicit schemes, which also implies that the implicit schemes are faster. For more information on time discretization schemes and on theoretical results with respect to stability, we refer to Quarteroni [QV08].

Generally, the time step size strongly depends upon the physical process which is simulated. For instance, in our MVR simulation scenario, shock waves are present, e.g., during the cardiac cycle, when the blood impulsively pushes onto the leaflets, or when contact is first established, and penalty terms just start to act. These sudden shocks make small time steps necessary, which may indicate the use of explicit codes. Yet, if according to the Courant criterion the time steps become too small in order to guarantee simulation stability, implicit methods will likely still be the better choice. Large time steps are usually sufficient when the response of a structure is governed by low frequency modes, like standard vibration problems.

Talking about time integration schemes in the context of FE models, it is also important to consider the effects of *numerical damping* which may remove spurious, non-physical high-frequency responses that are a feature inherent to the FEM. Thus, numerical damping may also lead to more stable convergence and allow for longer time steps and hence faster simulations.

In our MVR simulation application, we have implemented both an explicit and an implicit time integration algorithm: The classical explicit fourth-order single-step Runge Kutta (RK4) method, and the well-established implicit Newmark method. In the following, we will outline both of these methods, and we present respective numerical results in Chapter 8.

4.3.2.1 Explicit time integration – Runge Kutta (RK4) with adaptive time step size routine

For the classic Runge Kutta RK4 method, first we consider the following generic initial value problem: Find a function $y: I \rightarrow \mathbb{R}^n$ such that

$$\dot{y} = f(t, y) \quad \text{and} \quad y(t_0) = y_0, \quad (4.114)$$

where $I \subset \mathbb{R}$ is a compact interval with $t_0 \in I$ and a function $f: \Omega \subset \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^n$, where Ω is a domain with $(t_0, y_0) \in \Omega$ and $I \subset \pi_1(\Omega)$, where π_1 denotes the projection onto the first coordinate.

Following Rannacher [Ran13], the RK4 method is then given by the following iterative

scheme:

$$y_{n+1} = y_n + h_n \Phi(t_n, h_n, y_n), \quad \text{with } n \in \mathbb{N}, \quad (4.115)$$

where the increment function is given by

$$\Phi(t_n, h_n, y_n) = \frac{1}{6} k_1 + \frac{1}{3} k_2 + \frac{1}{3} k_3 + \frac{1}{6} k_4, \quad (4.116)$$

with

$$k_1 = f(t_n, y_n), \quad (4.117)$$

$$k_2 = f\left(t_n + \frac{h_n}{2}, y_n + \frac{h_n}{2} k_1\right), \quad (4.118)$$

$$k_3 = f\left(t_n + \frac{h_n}{2}, y_n + \frac{h_n}{2} k_2\right), \quad (4.119)$$

$$k_4 = f(t_n + h_n, y_n + h_n k_3). \quad (4.120)$$

In order to efficiently obtain a solution to a given initial value problem, and to therein guarantee that the error stays within user-specified bounds, the RK4 method should – in practice – be combined with an *adaptive time step size routine*, meaning that, given a tolerance $\tau_0 > 0$, a sequence of time step sizes h_n is to be constructed such that the error at $t = T$ is of the order of τ_0 . One way to do this is to determine the h_n such that for all time steps the truncation error satisfies

$$\|\tau(t_n, h_n, y_n)\| \approx h_n \tau_0, \quad (4.121)$$

thereby aiming to control the error per unit step.

Following, one can extrapolate the truncation error $\tau(t_n, h, y_n)$ as a function of h for a given point $(t_n, y_n) \in \Omega$ on the basis of an approximation of the function value $y(t_n + H)$ for a given trial step size $H > 0$.

Using the RK4 method we compute

$$\tilde{y} = y_n + H \Phi(t_n, H, y_n) \quad (4.122)$$

and

$$y'_n = y_n + \frac{H}{2} \Phi\left(t_n, \frac{H}{2}, y_n\right) \quad (4.123)$$

$$\hat{y} = y'_n + \frac{H}{2} \Phi\left(t_n + \frac{H}{2}, \frac{H}{2}, y'_n\right). \quad (4.124)$$

According to [Ran13], one has

$$\tau(t_n, h, y_n) \approx \frac{\tilde{y} - \hat{y}}{(1 - 2^{-p})H^{p+1}} h^{p+1}. \quad (4.125)$$

Consequently, using the step size $h = H$, we have the following estimate for the norm of the truncation error:

$$\text{EST} := \frac{\|\tilde{y} - \hat{y}\|}{1 - 2^{-p}} \approx \|\tau(t_n, H, y_n)\|. \quad (4.126)$$

Substituting equations (eq. 4.125) and (eq. 4.126) into equation (eq. 4.121), and solving for h , yields a rule for the choice of next time step's size:

$$h = \left(\frac{\tau_0}{\text{EST}} H^{p+1} \right)^{\frac{1}{p}}. \quad (4.127)$$

Suitable adaptive time step size routines are given, e.g., in [Ran13] or [Bat96]. In such an adaptive time step size algorithm, in order to increase the probability of the acceptance of a respectively new time step size, one may introduce a damping factor $\rho \in (0, 1]$ and a growth barrier $\eta \geq 1$. As the evaluation of the increment function Φ is usually very computationally expensive, one can minimize the number of necessary evaluations like this. – In our case one would have to solve four large linear systems, see equations (4.117) to (4.120) for the general case, or equations (4.130) to (4.133) for the case of the instationary contact elasticity problem in (eq. 4.113).

In order to now apply this generic formulation of the RK4 method with the adaptive time step size routine to the differential equations of continuum mechanics, i.e., to our MVR model equations, we first have to transform the instationary contact elasticity problem in (eq. 4.113) into an equivalent first-order system which corresponds to the generic form of the (eq. 4.114):

Therefore, let

$$\mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} := \begin{pmatrix} \mathbf{u} \\ \dot{\mathbf{u}} \end{pmatrix}. \quad (4.128)$$

Derivation of \mathbf{y} with respect to the time variable yields

$$\dot{\mathbf{y}} = \begin{pmatrix} \dot{\mathbf{y}}_1 \\ \dot{\mathbf{y}}_2 \end{pmatrix} = \begin{pmatrix} \dot{\mathbf{u}} \\ \ddot{\mathbf{u}} \end{pmatrix} = \begin{pmatrix} \mathbf{y}_2 \\ \mathbf{M}^{-1} (\mathbf{R} - \mathbf{S}(\mathbf{y}_1) - \mathbf{D}\mathbf{y}_2) \end{pmatrix} =: f(t, \mathbf{y}), \quad (4.129)$$

where we renamed the right-hand side vector $\mathbf{f}(t)$ into \mathbf{R} , and where \mathbf{S} denotes a linear or nonlinear functional, which in case of linear elasticity corresponds to the stiffness matrix \mathbf{K} as in (eq. 4.109).

Now, we can explicitly compute the k_i as in equations (4.117) to (4.120) for this first-order system; we denote the components of the k_i by k_{i1} and k_{i2} :

$$\begin{aligned} k_1 &= f(t_n, \mathbf{y}^n) \\ &= \begin{pmatrix} \mathbf{y}_2^n \\ \mathbf{M}^{-1} (\mathbf{R} - \mathbf{S}(\mathbf{y}_1^n) - \mathbf{D}\mathbf{y}_2^n) \end{pmatrix}; \end{aligned} \quad (4.130)$$

$$\begin{aligned} k_2 &= f\left(t_n + \frac{h_n}{2}, \mathbf{y}^n + \frac{h_n}{2} k_1\right) \\ &= \begin{pmatrix} \mathbf{y}_2^n + \frac{h_n}{2} k_{12} \\ \mathbf{M}^{-1} \left(\mathbf{R} - \mathbf{S} \left(\mathbf{y}_1^n + \frac{h_n}{2} k_{11} \right) - \mathbf{D} \left(\mathbf{y}_2^n + \frac{h_n}{2} k_{12} \right) \right) \end{pmatrix}; \end{aligned} \quad (4.131)$$

$$\begin{aligned}
k_3 &= f\left(t_n + \frac{h_n}{2}, \mathbf{y}^n + \frac{h_n}{2} k_2\right) \\
&= \left(\begin{array}{c} \mathbf{y}_2^n + \frac{h_n}{2} k_{22} \\ \mathbf{M}^{-1} \left(\mathbf{R} - \mathbf{S} \left(\mathbf{y}_1^n + \frac{h_n}{2} k_{21} \right) - \mathbf{D} \left(\mathbf{y}_2^n + \frac{h_n}{2} k_{22} \right) \right) \end{array} \right); \tag{4.132}
\end{aligned}$$

$$\begin{aligned}
k_4 &= f(t_n + h_n, \mathbf{y}^n + h_n k_3) \\
&= \left(\begin{array}{c} \mathbf{y}_2^n + h_n k_{32} \\ \mathbf{M}^{-1} \left(\mathbf{R} - \mathbf{S} \left(\mathbf{y}_1^n + h_n k_{31} \right) - \mathbf{D} \left(\mathbf{y}_2^n + h_n k_{32} \right) \right) \end{array} \right). \tag{4.133}
\end{aligned}$$

In order to complete the definition of the initial value problem as in the generic form of (eq. 4.114) we let

$$\mathbf{y}_1^0 := \mathbf{y}_1(t_0) = \mathbf{0} \quad \text{and} \quad \mathbf{y}_2^0 := \mathbf{y}_2(t_0) = \mathbf{0} \tag{4.134}$$

at the beginning of the simulation, i.e., there is no initial displacement and the initial velocity is zero.

Knowing that explicit time integration schemes in the context of the simulation of almost incompressible soft tissues usually are a poor choice, we will consider and implement an implicit method, too. As an example, the *implicit Euler* scheme is unconditionally stable and emerged as the de-facto standard for real-time deformable soft tissue simulation. However, it only has a convergence order of one, and in some cases it is desirable to achieve higher accuracy for the time discretization technique. An attractive alternative is given by the Newmark- β method, where for structural dynamics applications usually $\beta = 0.25$, which yields the *constant average acceleration scheme*. Its additional computational overhead is negligible compared to the implicit Euler scheme and it is of second order accuracy, see next paragraph.

4.3.2.2 Implicit time integration – Newmark scheme

Among many implicit integration schemes, the implicit Newmark method is the most well-known to solve continuum mechanics problems [Bat96, Wri06], such as the equation of motion (eq. 4.113). It is based on the following assumptions

$$\frac{{}^{t+\Delta t}\dot{\mathbf{u}} - {}^t\dot{\mathbf{u}}}{\Delta t} = (1 - \delta) {}^t\ddot{\mathbf{u}} + \delta {}^{t+\Delta t}\ddot{\mathbf{u}}, \tag{4.135}$$

$$\frac{{}^{t+\Delta t}\mathbf{u} - {}^t\mathbf{u}}{\Delta t^2} = \frac{{}^t\dot{\mathbf{u}}}{\Delta t} + \left(\frac{1}{2} - \beta\right) {}^t\ddot{\mathbf{u}} + \beta {}^{t+\Delta t}\ddot{\mathbf{u}}, \tag{4.136}$$

from which the subsequent approximations of velocities and displacements at time $t + \Delta t$ can be inferred:

$${}^{t+\Delta t}\dot{\mathbf{u}} = {}^t\dot{\mathbf{u}} + \left[(1 - \delta) {}^t\ddot{\mathbf{u}} + \delta {}^{t+\Delta t}\ddot{\mathbf{u}} \right] \Delta t, \tag{4.137}$$

$${}^{t+\Delta t}\mathbf{u} = {}^t\mathbf{u} + {}^t\dot{\mathbf{u}}\Delta t + \left[\left(\frac{1}{2} - \beta\right) {}^t\ddot{\mathbf{u}} + \beta {}^{t+\Delta t}\ddot{\mathbf{u}} \right] \Delta t^2. \tag{4.138}$$

Hence, the displacements and velocities depend upon values at time t , but also on the accelerations at time $t + \Delta t$. The implicit Newmark scheme is thus very robust with respect to oscillations, especially for applications using Rayleigh Damping. The parameters β and δ (with $0 \leq \beta \leq 0.5$ and $0 \leq \delta \leq 1$) can be determined and regulated to adapt integration order, accuracy and stability. Newmark originally proposed as an unconditionally stable scheme the *constant average acceleration method* (also called *trapezoidal rule*) in which case $\delta = \frac{1}{2}$ and $\beta = \frac{1}{4}$, see, e.g., Bathe [Bat96]. In addition to being unconditionally stable, this scheme also shows the highest asymptotical accuracy and consistency of the order 2. In our simulation application, we therefore stick to these values.

For the solution of the displacements, velocities and accelerations at time $t + \Delta t$, the Newmark methods also considers the equilibrium equation (eq. 4.113) at time $t + \Delta t$, which reads as

$$\mathbf{M} \, {}^{t+\Delta t}\ddot{\mathbf{u}} + \mathbf{D} \, {}^{t+\Delta t}\dot{\mathbf{u}} + \mathbf{K} \, {}^{t+\Delta t}\mathbf{u} = {}^{t+\Delta t}\mathbf{R}, \quad (4.139)$$

where we omitted the subscript h from space discretization for better readability, and again renamed the right-hand side vector into \mathbf{R} .

Solving from (eq. 4.138) for ${}^{t+\Delta t}\ddot{\mathbf{u}}$ in terms of ${}^{t+\Delta t}\mathbf{u}$, and then substituting for ${}^{t+\Delta t}\ddot{\mathbf{u}}$ into (eq. 4.137), we obtain equations for ${}^{t+\Delta t}\ddot{\mathbf{u}}$ and ${}^{t+\Delta t}\dot{\mathbf{u}}$, each in terms of the unknown displacements ${}^{t+\Delta t}\mathbf{u}$ only.

These two relations for ${}^{t+\Delta t}\ddot{\mathbf{u}}$ and ${}^{t+\Delta t}\dot{\mathbf{u}}$ are substituted into (eq. 4.139), which can then be solved for ${}^{t+\Delta t}\mathbf{u}$. Thereafter, using (eq. 4.137) and (eq. 4.138), ${}^{t+\Delta t}\ddot{\mathbf{u}}$ and ${}^{t+\Delta t}\dot{\mathbf{u}}$ can be calculated, too.

In the following, we list the complete algorithm which thus arises from the Newmark integration scheme, cf. [Bat96]. Note that the presented structure can also directly be found in the implementation of our MVR surgery simulation.

(A) Initial calculations:

(A.1) Form stiffness matrix \mathbf{K} , mass matrix \mathbf{M} , and damping matrix \mathbf{D} .

(A.2) Initialize ${}^0\mathbf{u}$, ${}^0\dot{\mathbf{u}}$, and ${}^0\ddot{\mathbf{u}}$.

(A.3) Select time step Δt and parameters β and δ , and calculate integration constants:

$$\begin{aligned} \delta &\geq 0.5; & \beta &\geq 0.25(0.5 + \delta)^2; \\ a_0 &= \frac{1}{\beta\Delta t^2}; & a_1 &= \frac{\delta}{\beta\Delta t}; & a_2 &= \frac{1}{\beta\Delta t}; & a_3 &= \frac{1}{2\beta} - 1; \\ a_4 &= \frac{\delta}{\beta} - 1; & a_5 &= \frac{\Delta t}{2} \left(\frac{\delta}{\beta} - 2 \right); & a_6 &= \Delta t(1 - \delta); & a_7 &= \delta\Delta t; \end{aligned}$$

(A.4) Form effective stiffness matrix $\hat{\mathbf{K}}$: $\hat{\mathbf{K}} = \mathbf{K} + a_0\mathbf{M} + a_1\mathbf{D}$.

(B) For each time step:

(B.1) Calculate effective loads at time $t + \Delta t$:

$${}^{t+\Delta t}\hat{\mathbf{R}} = {}^{t+\Delta t}\mathbf{R} + \mathbf{M}(a_0 {}^t\mathbf{u} + a_2 {}^t\dot{\mathbf{u}} + a_3 {}^t\ddot{\mathbf{u}}) + \mathbf{D}(a_1 {}^t\mathbf{u} + a_4 {}^t\dot{\mathbf{u}} + a_5 {}^t\ddot{\mathbf{u}}).$$

(B.2) Solve for displacements at time $t + \Delta t$:

$$\hat{\mathbf{K}} \mathbf{}^{t+\Delta t} \mathbf{u} = \mathbf{}^{t+\Delta t} \hat{\mathbf{R}}.$$

(B.3) Calculate accelerations and velocities at time $t + \Delta t$:

$$\begin{aligned} \mathbf{}^{t+\Delta t} \ddot{\mathbf{u}} &= a_0 (\mathbf{}^{t+\Delta t} \mathbf{u} - \mathbf{}^t \mathbf{u}) - a_2 \mathbf{}^t \dot{\mathbf{u}} - a_3 \mathbf{}^t \ddot{\mathbf{u}}, \\ \mathbf{}^{t+\Delta t} \dot{\mathbf{u}} &= \mathbf{}^t \dot{\mathbf{u}} + a_6 \mathbf{}^t \ddot{\mathbf{u}} + a_7 \mathbf{}^{t+\Delta t} \ddot{\mathbf{u}}. \end{aligned}$$

For linear problems, analysis of the Newmark method with respect to accuracy and stability can be found, e.g., in Hughes [MH83]. For parameters $\delta \leq 0.5$, the accuracy of Newmark's method is of the order $\mathcal{O}(\Delta t^2)$. Using parameters $\delta > 0.5$, one can damp out higher frequency responses, however, this leads to a reduction of the order of accuracy. Hence, specific methods have been developed which retain the order but still have the required damping properties, see, e.g., Hilber et al. [HHT77].

Concluding, we note that the linear system in step **B.2** is directly formulated in terms of the displacements, which – as we will see in the subsequent paragraph – allows to easily incorporate the above-mentioned displacement constraints, i.e., displacement boundary conditions.

4.3.2.3 Incorporation of Displacement Constraints (Dirichlet BCs)

Looking at the treatment of the boundary conditions in our MVR model, in the FE formulation in (eq. 4.113), we so far only consider the pressure and contact BCs, which are naturally included via the boundary integrals in (eq. 4.106) and thus through the corresponding assembly specification of the load vector in (eq. 4.111).

In contrast, fixed and displacement constraints – represented by Dirichlet BCs – are so far only prescribed by means of the solution space \mathbf{V}_h , with $\mathbf{u}_h \in \mathbf{V}_h$ such that $\mathbf{u}_h - \mathbf{u}_{h,D} \in \mathbf{V}_{h,0} = \mathbf{V}_h \cap \mathbf{H}_0^1(\Gamma_D; \Omega)$, see again (eq. 4.106). Thus, they have to be explicitly handled and imposed by means of a specific procedure.

This being said, when using the above-presented RK4 or Newmark time integration scheme, the resulting linear systems are directly formulated in terms of the displacements, which enormously simplifies this task: We will see that displacements at element nodes can be incorporated in a very computationally efficient way, however, it is rather difficult to prescribe displacement constraints at completely arbitrary points of the body [BLME13]. In the following, we first present the standard procedure in the FE context to achieve the incorporation of Dirichlet BCs for element nodes, and will get back to the issue of prescribing displacements at arbitrary points afterwards.

When the linear system is formulated in terms of the displacements, imposing that $\mathbf{u} = \mathbf{u}_D$ on Γ_D corresponds to setting the DoFs u_i to the respectively prescribed values $u_{D,i}$, if these lie on this boundary part. One way of doing this is to modify the linear system that arises from the space and time discretization: For each (boundary) DoF u_i , we need to replace the equation containing u_i by the equation $u_i = u_{D,i}$. This can be done by simply setting the i -th row of the system matrix to the identity row, with 0 everywhere except in position i , where we put 1, and the i -th component of the right-hand side vector is set to $u_{D,i}$. Note that the elimination

of the i -th column is not necessarily needed, which is acceptable as long as an iterative method is used to solve the system. In this case, it just needs to be assured that the initial guess in the iterative method does contain the value $u_{D,i}$ of the boundary DoF u_i , since this value will then be conserved throughout the iterative process.

Mathematically, this procedure is known as the *displacement projection method*: For a prescribed displacement $u_{D,i}$ of certain nodes i in the set \mathcal{N} , the dimension of the linear system that results from the discretization process is essentially reduced by the size of \mathcal{N} . However, in order to conserve matrix symmetry, usually one does not change the size of the linear system, but instead builds the displacements into the system by means of the above described procedure. The core idea is hence to *project* the nodal displacements to the given values, i.e., $u_i = u_{D,i}$ for all $i \in \mathcal{N}$, and to then modify the linear system such that for the result $du_i = 0$ for all $i \in \mathcal{N}$.

Please note that in the instationary case, of course, the prescribed displacement values of the respective nodal points grow (linearly) over time, i.e., we impose $\mathbf{u}(\mathbf{x}, t) = \frac{t}{T} \mathbf{u}_D(\mathbf{x}) = \frac{n\Delta t}{T} \mathbf{u}_D(\mathbf{x})$, where t denotes the current time, n the current time step, and T is the simulation time at the end of which the respective displacement shall be obtained. For more details, we refer back to Section 4.2.1.2 where we gave thorough information and schematic visualization of our way of modeling pointwise Dirichlet BCs.

The displacement or fixation of arbitrary points on the boundary or inside a body by means of Dirichlet BCs is a little more difficult. Belytschko describes problems and solution methods for this issue [BLME13]. In the case of our MVR simulation, we handle arbitrary points that shall be subjected to displacement or fixation in the following way:

First, for an arbitrary point we check which (tetrahedral) element this specific point is contained in, and then determine this element's closest surface facet to the specific point. By definition, we then impose the Dirichlet BCs, i.e., the displacement or fixation, to all nodes of this element facet. For a sufficiently and model-related adequately refined mesh the simulated effects of this representation physically get very close to the real displacement of the real point. With the thus obtained set of element facet nodes – that correspond to the DoFs u_i – we proceed as described above.

4.3.2.4 Outlook and Transition (to the FEM implementation)

Now, that we have derived a way to numerically solve the above specified contact elasticity problem for the MV and for MVR in Figure 4.11 by means of the Finite Element Method (FEM), we can have a look at the implementation of the respective FEM-based simulation in the next Chapter.

As mentioned before, with the derived simplistic biomechanical MV/MVR model we aim at investigating on the *integration* of an MV/MVR simulation into a *cognitive surgery assistance system*, rather than at the presentation of a further advanced model. Concerning the implementation, we will hence put particular emphasis on a generic and flexibly extensible structure that serves as a simulation prototype for research purposes. Accordingly, we will also focus on competent and flexible *simulation control mechanisms and simulation interfaces* that facilitate the setup of patient-specific and surgical expert knowledge-based simulation scenarios.

5 MVR simulation using the FEM

Having motivated and derived a way to numerically solve the MVR contact elasticity problem by means of the Finite Element Method (FEM) in the previous chapter, in this chapter, we will focus on the implementation of the respective FEM-based simulation application.

Therefore, first, we have a look at the structure of a general FEM software application code, building up on the EMCL inhouse-developed FEM software toolkit HiFlow³, see Section 5.1. Second, we will extend this general structure in order to incorporate the particular MVR-related features for our specific MV/MVR simulation application, see Section 5.2. In doing so, we will put particular emphasis on a generic and extensible code structure as well as on competent and flexible simulation control mechanisms and interfaces which facilitate the consideration of patient-specific medical data and surgical expert knowledge in order for a cognitive integration into a cognition-guided cardiac surgery assistance system.

5.1 General Setup of an FEM-based Simulation

In this section, we investigate on the general setup of an FEM software application code. We discuss the essential requirements for simulation applications, and look at the FEM software toolkit HiFlow³, which has been developed in the Engineering Mathematics and Computing Lab (EMCL), see 5.1.1. Following, in 5.1.2, we then focus on the basic structural building blocks of an FEM-based simulation code.

5.1.1 FEM Software HiFlow³

FEM-based numerical simulations of physical processes that can be described by means of partial differential equations (PDEs) have become an important tool in engineering, research and development.

They can virtually reproduce, i.e., *simulate*, the behavior of physical objects, and thus allow for replacing costly experiments and studies, or even for optimization of products or processes. However, complex geometries of the objects simulated, high accuracy requirements and the need for short simulation times constitute big challenges, still today. For this reason, it is imperative to maximally exploit modern high performance computing infrastructure, and to optimize the simulation algorithms with respect to the specific simulation application and the underlying numerics.

For this purpose, there is a number of open-source as well as commercial FEM software toolkits and libraries, which were already mentioned in Chapter 3.1.5.1. Well-established open-source simulation engines are, for example, deal.II [BHH⁺16], Dune [DKNO10], HiFlow³ [AAB⁺12] and SOFA [FDD⁺12]. Among the commercially available FEM simulation software packages, one will find Abaqus, Ansys, COMSOL Multiphysics, FEBio, Matlab, Materialise Mimics and 3matics, and many more. Besides these software packages, there is a number of

useful libraries which are, for instance, dedicated to solving linear systems that arise from FE discretization, e.g., PETSc [BBE⁺08] or Hypr [FJY06]. *PETSc* is a highly scalable linear algebra library which provides data structures and routines for large-scale scientific applications that are generally modeled by means of PDEs. Their mechanism are optimized for parallel application codes, and include, for example, parallel matrix and vector assembly routines as well as linear solving routines, which allows the user to keep control over the entire solution process. Similarly, the parallel High Performance Preconditioners *Hypr* is a library of routines for the scalable solution of linear systems. The main strength of *Hypr* is the availability of high performance parallel multigrid preconditioners for both structured and unstructured grid problems.

In our work, we make use of our EMCL inhouse-developed, open-source C++ software HiFlow³ [AAB⁺12]. HiFlow³ is a multi-purpose FE software toolkit which provides powerful tools for an efficient and accurate solution of a wide range of large-scale coupled problems modeled by PDEs. According to the common structure of an FEM software, in order to achieve maximal flexibility and applicability, it follows a *modular and generic setup* approach for building efficient parallel numerical solvers, and it provides highly capable modules that deal with the mesh setup, finite element spaces, degrees of freedom, linear algebra routines, and output data for visualization. Built upon object-oriented concepts and the full capabilities of C++, HiFlow³ aims at the full utilization of available computing resources on heterogeneous platforms – from stand-alone workstations to large HPC clusters, from multi-core CPUs to multi-GPUs. Its three core modules (Mesh, DoF/FEM, and the Linear Algebra toolbox) are therefore *intrinsically parallel*: an MPI-layer realizes the communication on distributed grids and distributed data structures, and a *hardware-aware* computing concept includes cross-platform-optimized, scalable linear algebra back-ends. The three core modules are supplemented with other standard building blocks that are typically part of every FEM-based simulation procedure, e.g., routines for numerical integration (quadrature or cubature) of element integrals, assembling of system matrices and vectors, inclusion of boundary conditions, the setup of linear and nonlinear solvers along with suitable preconditioners, error estimators, and output visualization tools.

For more information on HiFlow³, its specific features and functionalities, and its large range of application use cases, we refer to the project website www.hiflow3.org and to the following literature: For a short high-level overview, please see [AAB⁺12]. For a detailed description of the design, the modules, the assembly concept, etc. and a demonstration of show cases and sample challenges along with a performance evaluation, we refer to [AAB⁺10]. Finally, a number of dedicated work in the context of specific use cases can be found in the different issues of the Preprint-Series of the Engineering Mathematics and Computing Lab (EMCL), see <http://journals.ub.uni-heidelberg.de/index.php/emcl-pp/issue/archive/>.

5.1.2 General FEM Code Structure

Following, we will have a look at the general setup and structure of an FEM-based simulation application, thereby referring to HiFlow³ Tutorials that are available as documentation on the HiFlow³ project website www.hiflow3.org. Thereon-based, we want to then proceed in the next section with a specification and adaptation of this general setup in order to implement our MV/MVR simulation application code.

Generally, an FEM simulation code has to facilitate miscellaneous tasks, which are usually taken care for in the framework of separated routines that provide the respective functionalities:

- **run()** to access, start and run the whole program.
- **read_parameters()** to read in the needed parameters, and to thus act as a control interface for simulation properties and simulation execution.
- **read_partition_and_distribute_mesh()** to read in the input geometry of the manipulated body, to then possibly refine the body's mesh representation, and to distribute it to the available compute infrastructure for parallel computation.
- **prepare_system()** to define and initialize the FE space and the Linear Algebra setup (possibly along with the respective communication features according to the above distributed mesh), to set the parameters in the principal modules and in the general numerics building blocks, and to initialize or prepare the following items:
 - **linear system** according to the respective problem formulation, cf. (eq. 4.113), such that the system matrix, the solution vector and the right hand side vector are initialized and available for further computation.
 - **boundary condition structures** according to the respective problem formulation, such that it can be employed in the **assemble_system()** routine.
 - **time stepping scheme** according to the respective problem formulation, such that the **run()** routine respectively facilitates a possible loop over the number of time steps.
- **assemble_system()** to compute and assemble the system matrix and the right hand side vector according to the respective problem formulation, thereby incorporating the respective time integration scheme and boundary conditions. Note, that this routine is possibly called in every time step in case of an instationary simulation.
 - **ElasticityAssembler-class** to facilitate the assembly of the system matrix **lm** (which in case of a general elasticity formulation includes mass, damping and stiffness terms) and of the right hand side vector **lv** (which incorporates the boundary conditions); see Section 4.3.2.1 or 4.3.2.2.


```

          * void operator()(const Element& element, const Quadrature<double>& quadrature,
            LocalMatrix& lm);

          * void operator()(const Element& element, const Quadrature<double>& quadrature,
            LocalVector& lv);

          * std::vector<double> compute_pressure_boundary_terms(int q, double pN);

          * std::vector<double> compute_contact_boundary_terms(int q, double pC);
          
```
 - **Dirichlet-BC facilitator** to compute the respective Dirichlet boundary values and DoFs, and to modify the DoFs of system matrix, solution vector and right hand side vector accordingly; see Section 4.3.2.3.

- `solve_system()` to solve the above assembled linear system (or, in case of nonlinear elasticity, respectively, its stepwise linearized equations according to, e.g., the Newton-Raphson Method), in order to obtain the displacement/solution vector; see Section 4.3.2.1 or 4.3.2.2. Note, that this routine is possibly called in every time step in case of an instationary simulation, and additionally in every linearization step for nonlinear elasticity.
- `visualize(int timestep)` to write out the solution and respectively relevant additional information for visualization and other post-processing purposes. Note, that this routine is possibly called in every time step in case of an instationary simulation.

Besides these basic routines, an FEM simulation code requires for an access and input control interface, which – for HiFlow³ – is composed by an `.xml` file and, respectively, an `.xml`-parser as part of HiFlow³. This `.xml` parameter file not only includes important problem-related parameters, but also specifies the numerics setup of the simulation (e.g., solvers, preconditioners, time integration scheme, tolerance thresholds, computing platform) and additionally may link mesh and boundary condition input files. For an example, see Listing 5.1:

Listing 5.1: Sample `.xml` parameter inputfile of the HiFlow³ elasticity simulation application.

```

1  <Param>
2  <Application>
3  <Elasticity>SoftTissueDeformation</Elasticity>
4  </Application>
5  <Input>
6  <Mesh>
7  <Filename>soft_body.inp</Filename>
8  <RefinementLevel>3</RefinementLevel>
9  <MaterialParameters>healthy</MaterialParameters>
10 </Mesh>
11 <BoundaryConditions>
12 <BCfilename>Dirichlet_and_Neumann.bcdata.xml</BCfilename>
13 </BoundaryConditions>
14 </Input>
15 <Output>
16 <Mesh>
17 <PathAndFilename>SimResults/elasticity_sim_ts</PathAndFilename>
18 </Mesh>
19 <LogFilename>instationary_elasticity_sim_info_log</LogFilename>
20 <DebugFilename>instationary_elasticity_sim_debug_log</DebugFilename>
21 </Output>
22 <LinearAlgebra>
23 <NameMatrix>CoupledMatrix</NameMatrix>
24 <NameVector>CoupledVector</NameVector>
25 <Platform>CPU</Platform>
26 <Parallelization>128</Parallelization>
27 <Implementation>Naive</Implementation>
28 <MatrixFormat>CSR</MatrixFormat>
29 </LinearAlgebra>
30 <FiniteElements>
31 <DisplacementDegree>1</DisplacementDegree>
32 </FiniteElements>
33 <Instationary>
34 <Method>Newmark</Method>

```

```

35     <Damping>1</Damping>
36     <MaxIts>10000</MaxIts>
37     <DeltaT>0.01</DeltaT>
38 </Instationary>
39 <LinearSolver>
40     <SolverName>CG</SolverName>
41     <PreconditioningMethod>SymmetricGaussSeidel</PreconditioningMethod>
42     <MaxIterations>2000</MaxIterations>
43     <AbsTolerance>1.e-14</AbsTolerance>
44     <RelTolerance>1.e-8</RelTolerance>
45 </LinearSolver>
46 </Param>

```

For more general and introductory information, we refer to our tutorial for HiFlow³-based soft tissue simulation, see the HiFlow³ Elasticity Tutorial [SKH15], and the tutorial for time integration schemes, see the HiFlow³ Convection Diffusion Tutorial [GSH14].

5.2 Setup of our FEM-based MVR Elasticity Simulation

Building up on the previous section, following, we will extend and specify the above general setup of an FEM-based simulation algorithm for the particular case of an MVR simulation application. The usage of biomechanical simulations in surgery assistance environments requires high simulation reliability and accuracy, almost real-time performance, and – besides that – the integration of patient-individual medical data as well as the consideration of surgical expert knowledge in model and simulation setup. We can only reliably address these issues in our simulations, when implementing a highly specific and performance-optimized FEM simulation code, that can make efficient use of high performance computing (HPC) resources, and that allows – via a competent and flexible simulation interface – for integrating surgical expert knowledge and patient-specific input data into respective simulation scenarios.

Extending our work [SES⁺15], in 5.2.1, we will describe the specific structure of our FEM-based MVR simulation application and its important features as well as the simulation input and control interfaces. In 5.2.2, we will focus on the numerics- and HPC-based optimization of our simulation code with respect to performance and accuracy. Finally, in 5.2.3, we give an outlook to the next two chapters, which will describe how – based on our simulation code setup and its interfaces – we allow for patient-specific and knowledge-based MVR simulations.

5.2.1 Analysis of the Code Structure, Code Control Mechanisms & Interfaces

In our MVR simulation application, we principally stick to the overall setup and structure of the above listed building blocks of a general FEM simulation algorithm, see again 5.1.2. These facilitate all essential tasks and provide the respective functionalities of an FEM application such as mesh setup, system assembly, etc.

In the following, we will describe the specifications and extensions which we have made for our MVR simulation. For more details, we refer to our work [SES⁺15], as well as to our source code and its thorough documentation which includes detailed comments and explanations.

- **run()**: This routine starts and runs the MVR simulation application for a specific MVR simulation scenario, e.g., through the command line interface. A specific `.xml` parameter

file that is handed over as an argument in the command line acts as MVR simulation scenario description, see below. The `run()` routine then calls the below routines one after the other or in loops and thus manages the entire simulation flow.

- **read_parameters()**: This routine reads in the parameters provided by the `.xml` parameter file, and then sets the simulation properties accordingly. It thus acts as a control interface for the simulation execution. Please find detailed explanations on the `.xml` parameter file below.
- **read_partition_and_distribute_mesh()**: A `.inp` or `.vtk` mesh representation of the MV morphology, which is linked in the `.xml` parameter file, is read in, possibly refined, and then distributed to the available compute infrastructure for subsequent parallel computation. According to the functionalities of HiFlow³, we either use the internal naive domain decomposition algorithms or those provided through the interface to the well-known parallel graph partitioning library METIS [KK98, ARK06].
- **prepare_system()**: Here, according to the `.xml` parameter file, we define and initialize the following numerical methods, structures, properties and settings:
 - **the Linear Algebra setup**: We initialize the Linear Algebra setup in order to consider, e.g., the above specified decomposed/distributed mesh and the thus given communication requirements (such as couplings and ghost cells), or HPC platform specifications, etc.
 - **the FE space**: We employ linear tetrahedral elements and an FE degree of first or second order for the displacement variable \mathbf{u} .
 - **the time stepping scheme**: The type of time integration scheme strongly depends on the respective problem formulation and the characteristics of the solution. We either use RK4 or Newmark as described in Section 4.3.2. The scheme's parameters and instances are set such that the `run()` routine accordingly facilitates the loop over the number of time steps.
 - **the material parameters**: The tissue properties, as given in the `.xml` parameter file, are initialized for the assembly description.
 - **the boundary condition structures**: According to the problem formulation (eq. 4.113), we initialize the set of boundary vectors which account for the prescribed time-dependent boundary conditions. The actual computation (*assembly*) of these boundary vectors then takes place in the `assemble_system()` routine.
 - **the linear system**: According to problem size, time integration scheme, and contact elasticity problem formulation (eq. 4.113), we initialize the system matrix, the solution vector, the right hand side vector, and all respectively required helper vectors and matrices, such that these are available for further computation.
 - **the solver setup**: Considering the specific characteristics of the contact elasticity problem formulation (eq. 4.113), we choose and initialize solver and preconditioner, thereby including, e.g., threshold values for iteration numbers, and tolerances, etc. We will investigate on possible optimal solver/preconditioner combinations with respect to our problem (eq. 4.113) below in Section 5.2.2.

- **assemble_system()**: The set of sub-routines that is activated by the **assemble_system()** routine is concerned with the computation and assembly of the system matrix and the right hand side vector according to the respective contact elasticity problem formulation (eq. 4.113) along with the prescribed boundary conditions. As we investigate on the dynamic behavior of the MV, we run an instationary simulation and thus apply a time integration scheme, such that the entire assembly routine is called in every time step. For a first overview over the whole assembly part of the MVR simulation code, see the pseudocode Listing 5.2. The therein-mentioned sub-routines of the **assemble_system()** routine are thoroughly explained in the following.
 - **ElasticityAssembler-class**: First, we prepare and compute the components of the boundary vectors on the one hand, and of the stiffness, mass, and damping matrices on the other hand, compare (eq. 4.113), respectively referring to (4.109), (4.110) and (4.111). This is to then facilitate – in the end of each iteration step of the time loop – the assembly of the linear system matrix and of the right hand side vector, e.g., according to the Newmark algorithm. Therefore, we consider the following sub-items:
 - * **the pointwise Neumann BCs**: We process the pointwise Neumann BCs (NBCs) as given in the `.xml` parameter file into the pointwise NBC data structure with respect to the given FE mesh. Thereon-based, the data structure then computes the direction and scaling of the chordae pull forces, thus imitates the elastic back-pulling effects of the chordae, see again Section 4.2.1.1. The respectively computed directions and scaling factors are assembled into the *pointwise NBC boundary vector*.
 - * **the surfacewise Neumann BCs**: We process the surfacewise NBCs as defined in the `.xml` parameter file into the surfacewise NBC data structure with respect to the given FE mesh. Thereon-based, the data structure then facilitates the computation of the time- and place-dependent directions and scaling factors, thus imitating the blood pressure throughout the cardiac cycle, compare to Section 4.2.1.1. The respectively computed directions and scaling factors are then assembled into the *surfacewise NBC boundary vector*.
 - * **the surfacewise Contact BCs**: We analyze the potential event of contact for all element facets on the upper leaflet side as described in Section 4.2.1.1. It should be noted, that the search for potential contact occurrences can enormously be accelerated, and we will put particular emphasis on this issue in Section 5.2.2. Once the contact facets are found, they are processed into a respective contact BC (CBC) data structure, in order to assemble the *surfacewise CBC boundary vector*.
 - * **the stiffness matrix components**: We compute the components of the stiffness matrix \mathbf{K} according to (eq. 4.109). It is important to mention, that the usage of linear FE basis functions (as opposed to quadratic ones) has a clear disadvantage from the convergence and accuracy point of view. However, it has a prominent advantage with respect to computation efficiency, since – due to $\nabla\phi$ being constant – the stiffness matrix is constant on the whole element.

- * **the mass matrix components:** We compute the components of the mass matrix \mathbf{M} according to (eq. 4.110).
- * **the damping matrix components:** In order to obtain the components of the damping matrix \mathbf{D} , we compute the linear combination of mass matrix \mathbf{M} and stiffness matrix \mathbf{K} , according to the approach of Rayleigh, see (eq. 4.56).
- **Dirichlet-BC-facilitator:** Here, we compute the mesh-related Dirichlet boundary DoFs and values, as derived in Sections 4.2.1.1 and 4.2.1.2, with respect to the natural cardiac cycle or to the surgical procedure of an annuloplasty. Once computed, we facilitate the corresponding modifications of system matrix, solution vector and right hand side vector as described in Section 4.3.2.3.
- **Assembly-to-time-integration-scheme-adapter:** The hence computed components of the above listed matrices and boundary vectors need to be combined with respect to the chosen time integration scheme, in order to then yield the overall system matrix and the overall right hand side vector. Using, e.g., the Newmark time integration method, we follow the rules given in steps (A.4) and (B.1) in Section 4.3.2.2. Please note that, assuming linear elasticity and when using linear FE basis functions, according to the Newmark algorithm, the effective stiffness matrix (or: system matrix) only needs to be assembled once at the beginning of the entire simulation in step (A.4) of the Newmark algorithm. Apart from achieving weaker convergence (which is mainly due to $\nabla\phi$ being constant), this is a remarkable performance advantage.
- **solve_system():** Having assembled the system matrix and the right hand side vector for the respective time step (or, in case of nonlinear elasticity, its stepwise linearized equations according to, e.g., the Newton-Raphson Method), we may now solve this system in order to obtain the displacement/solution vector. The application of a suitable preconditioner along with the choice of an appropriate solver which fit to the respective system properties may enormously accelerate the solving process; we will give details on this issue below in Section 5.2.2. Having solved the system, we update the solution and helper vectors according to the respective time integration scheme, see, e.g., step (B.3) in the Newmark algorithm in Section 4.3.2.2.
- **visualize(int timestep):** At the end of each time step, we write out the solution vector, which describes the current deformation of the MV, and compute the so-called *von Mises stress*, which is an important physical measure and of high clinical relevance. In a simulation post-processing step, the deformation and the von Mises stress values can be visualized and thus allow the surgeon for easy interpretation and surgical assessment.

At this point, we remark, that the implementation and the overall design of our MVR soft tissue simulation application is to maximally exploit the highly capable modules of HiFlow³. In addition to that, we made an effort to optimally handle and parallelize all features related to the specific contact elasticity problem formulation (eq. 4.113). We will focus on numerics and HPC-related code optimization below, see Section 5.2.2.

Generally, according to Sections 4.2.1.1 and 4.2.1.2, we subdivide an MVR surgery simulation scenario into two parts: Part 1 describes the annuloplasty ring implantation procedure. Part 2

describes a post-MVR-surgery cardiac cycle, where the valve is subjected to the constraints induced by the implanted ring prosthesis.

In part 1, we strictly follow standard clinical guidelines of ring annuloplasty surgery as explained in Section 2.2. The respective model has been described in Section 4.2.1.2. A disease-specific ring prosthesis is implanted by imposing appropriate displacements on the MV annulus, and the manipulated valve is then left alone for a short while in order to damp the implantation-induced oscillations. In part 2, a physiological transvalvular pressure gradient (of up to 120 mmHg) is applied to the ventricular side of the MV leaflets throughout an artificially extended 10 s timeframe. A simulation of the MV function is thus performed for the closing part of the cardiac cycle or even for an entire cardiac cycle.

The two parts are triggered by means of respectively time-dependent boundary conditions.

Listing 5.2: Structure of the assembly and solver part of our MVR soft tissue simulation code.

```

1  for all timesteps do
2    assemble_system() {
3      assemble_boundary_vectors() {
4        process fixed displacement boundary elements (i.e., annuloplasty)
5        process pointwise force boundary elements (i.e., chordae)
6        process facetwise force boundary elements (i.e., blood pressure)
7        analyze potential contact in current timestep (i.e., leaflet contact)
8        process contact elements
9      }
10     assemble_matrices() {
11       compute mass matrix components
12       compute stiffness matrix components
13     }
14     adapt_assembly_to_time_integration_scheme() {
15       assemble right-hand-side vector
16       assemble system matrix
17     }
18   }
19   solve_system() {
20     setup and apply preconditioner
21     execute solver
22   }
23   update_solution_vectors() {
24   }
25 end time loop

```

The Simulation Scenario Input and Control Interface. As mentioned in the list of routines above, and as described in Section 5.1.2, too, we make use of an `.xml` parameter file in order to describe and control the respective MVR simulation scenario setup. In order to have this `.xml`-file act as our general simulation control interface, we have its contents parsed by means of the HiFlow³-`.xml`-parser, which we adapted for MVR surgery simulations for our HiFlow³-based FEM soft tissue simulation code. Listing 5.3 shows the sample contents of a template of a such `.xml` parameter file. It not only includes important problem-related parameters, but also specifies the numerics setup of the simulation (e.g., solvers, preconditioners, time integration scheme, tolerance thresholds, hardware platform, etc.) and additionally links the input mesh and possibly multiple boundary condition input files.

Listing 5.3: Sample .xml parameter input file of our MVR soft tissue simulation application.

```

1  <Param>
2  <ElasticityApplication>MVRsimulation</ElasticityApplication>
3  <SimulationInput>
4    <Mesh>
5      <Filename>MVmesh.inp</Filename>
6      <RefinementLevel>2</RefinementLevel>
7      <MaterialParams>healthy</MaterialParams>      <!-- healthy or sick -->
8    </Mesh>
9    <BoundaryConditions>
10     <BCfilename1>DirichletBCs_Annuloplasty.bcdata.xml</BCfilename1>
11     <BCfilename2>NeumannBCs_Chordae.bcdata.xml</BCfilename2>
12     <BCfilename3>NeumannBCs_BloodPressure.bcdata.xml</BCfilename3>
13     <BCfilename4>ContactBCs_LeafletContact.bcdata.xml</BCfilename4>
14   </BoundaryConditions>
15 </SimulationInput>
16 <SimulationOutput>
17   <Mesh>
18     <PathAndFilename>SimResults/elasticity_sim_ts</PathAndFilename>
19   </Mesh>
20   <LogFilename>MVR_instat_elast_sim_info_log</LogFilename>
21   <DebugFilename>MVR_instat_elast_sim_debug_log</DebugFilename>
22 </SimulationOutput>
23 <LinearAlgebra>
24   <NameMatrix>CoupledMatrix</NameMatrix>
25   <NameVector>CoupledVector</NameVector>
26   <Platform>CPU</Platform>      <!-- CPU or GPU -->
27   <Parallelization>128</Parallelization>      <!-- num parallel proc -->
28   <Implementation>Naive</Implementation>
29   <MatrixFormat>CSR</MatrixFormat>
30 </LinearAlgebra>
31 <FiniteElements>
32   <DisplacementDegree>1</DisplacementDegree>
33 </FiniteElements>
34 <QuadratureOrder>2</QuadratureOrder>
35 <Instationary>
36   <Method>Newmark</Method>
37   <Damping>1</Damping>
38   <RayleighAlpha>0.5<RayleighBeta>
39   <RayleighBeta>0.5<RayleighBeta>
40   <MaxIts>10000</MaxIts>
41   <DeltaT>0.01</DeltaT>
42   <VisOutputEvery1inXtimesteps>100</VisOutputEvery1inXtimesteps>
43 </Instationary>
44 <LinearSolver>
45   <SolverName>CG</SolverName>
46   <MaxIterations>2000</MaxIterations>
47   <Preconditioner>SymmetricGaussSeidel</Preconditioner>
48     <!-- SymmetricGaussSeidel or HypreAMG -->
49   <AbsoluteTolerance>1.e-14</AbsoluteTolerance>
50   <RelativeTolerance>1.e-8</RelativeTolerance>
51 </LinearSolver>
52 </Param>

```

The boundary condition files linked in lines 12 to 15 have the following structure:

The file named **DirichletBCs_Annuloplasty.bcddata.xml** describes the positions on the MV where the fixation or prescribed displacement takes place, cf. Sections 4.2.1.1 and 4.2.1.2, and respectively what the displacement vector looks like:

```

1 <Param><BCData>
2   <DirichletConstraints>
3     <NumFixedDBCpoints> $n_f$ </NumFixedDBCpoints>
4     <FixedDBCpoints>(x1, y1, z1); (x2, y2, z2); ...; (x $n_f$ , y $n_f$ , z $n_f$ )</FixedDBCpoints>
5     <NumDisplacementDBCpoints> $n_d$ </NumDisplacementDBCpoints>
6     <DisplacementDBCpoints>(x1, y1, z1); ...; (x $n_d$ , y $n_d$ , z $n_d$ )</DisplacementDBCpoints>
7     <DisplacementVectors>(dx1, dy1, dz1); ...; (dx $n_d$ , dy $n_d$ , dz $n_d$ )</DisplacementVectors>
8   </DirichletConstraints>
9 </BCData></Param>

```

The file named **NeumannBCs_Chordae.bcddata.xml** describes the positions on the leaflets where the Chordae tendinae are attached, and the respective directions into which these attachment points are pulled during the closure process of the MV; cf. Section 4.2.1.1. Along with this information, a scalar pulling factor **ChordaePullStrength** calibrates the chordae's elastic behavior:

```

1 <Param><BCData>
2   <NeumannConstraints_pointwise>
3     <NumChordaeNBCs> $n_{ch}$ </NumChordaeNBCs>
4     <ChordaeNBCpoints>(x1, y1, z1); ...; (x $n_{ch}$ , y $n_{ch}$ , z $n_{ch}$ )</ChordaeNBCpoints>
5     <ChordaePullVectors>(dx1, dy1, dz1); ...; (dx $n_{ch}$ , dy $n_{ch}$ , dz $n_{ch}$ )</ChordaePullVectors>
6     <ChordaePullStrength>0.8</ChordaePullStrength>
7   </NeumannConstraints_pointwise>
8 </BCData></Param>

```

The file named **NeumannBCs_BloodPressure.bcddata.xml** describes the element facet or surface identifiers (in HiFlow³: the material numbers), which indicate the upper or lower sides, and the anterior or posterior parts of the MV leaflets, in order to have the respective blood pressure profiles act on them according to Section 4.2.1.1. Along with the material numbers, we give information on the pressure scaling (no pressure [0], low [1] or high [2] blood pressure), the pressure timing (starting time of the post-MVR-surgery cardiac cycle in the simulation time scale) and the pressure direction (towards the annulus center point and center line):

```

1 <Param><BCData>
2   <NeumannConstraints_surfacewise>
3     <NBC_MatNr_AntLeafletUp>17</NBC_MatNr_AntLeafletUp>
4     <NBC_MatNr_PostLeafletUp>18</NBC_MatNr_PostLeafletUp>
5     <NBC_MatNr_LeafletLow>20</NBC_MatNr_LeafletLow>
6     <NBC_Scaling_LeafletUp>1</NBC_Scaling_LeafletUp>
7     <NBC_Scaling_LeafletDown>0</NBC_Scaling_LeafletDown>
8     <NBC_StartingTime>10</NBC_StartingTime>
9     <NBC_AnnulusRingRadius>2.47</NBC_AnnulusRingRadius>
10    <NBC_AnnulusCenterPoint>(xc, yc, zc)</NBC_AnnulusCenterPoint>
11  </NeumannConstraints_surfacewise>
12 </BCData></Param>

```

The file named `ContactBCs_LeafletContact.bcddata.xml` describes the element surface (facet) identifiers (the so-called material numbers in HiFlow³), which indicate the anterior or posterior MV leaflets, in order to have the contact analysis (i.e., the search for potential contact) accordingly facilitated; cf. Section 4.2.1.1. Moreover, we give the information on the minimally tolerated threshold gap distance $d_{\text{gap}} = \varepsilon_{\text{tol}}$, as from which on we define contact to be established. We then let the penalty term p_C grow towards infinity just before this gap distance shrinks towards 0 mm, thereby using the `PenaltyFactor` to scale the growth of this penalty term:

```

1 <Param><BCData>
2   <ContactConstraints>
3     <CBC_MatNr_AntLeafletUp>17</CBC_MatNr_AntLeafletUp>
4     <CBC_MatNr_PostLeafletUp>18</CBC_MatNr_PostLeafletUp>
5     <CBC_ToleranceThreshold>0.7</CBC_ToleranceThreshold>
6     <CBC_PenaltyFactor>10</CBC_PenaltyFactor>
7   </ContactConstraints>
8 </BCData></Param>

```

The Simulation Scenario Output. The output of our FEM-based MVR soft tissue simulation application is constituted by three different types of files:

Firstly, we produce a **simulation log file**, which records the important characteristics of the simulation execution. On the one hand, these consist of numerical properties, such as solver/preconditioner settings, iteration numbers, accuracy achievements, the time requirements for the different steps of the algorithm and time stepping scheme, etc. On the other hand, they also include surgically relevant parameters, such as the computed size of the leaflet coaptation zone for the different time steps of the simulated cardiac cycle.

Secondly, we obtain an **error and debug log file** in case there are any errors or potential problems, which may distort the simulation results, or which could even terminate the simulation execution. The respective log data is particularly important for further development works with respect to model extension and simulation enhancement.

Lastly, when the simulation is executed in parallel, the actual **simulation scenario results** consist – for every time step \mathbf{ts} – of one `.pvtu` file and several (n_{proc}) `.vtu` files. The n_{proc} different `.vtu` files originate from the parallel computations on the n_{proc} different compute nodes, every one of which produces an own `.vtu` file which contains a respective part (*domain*) of the current time step’s deformed MV geometry. The `.pvtu` file then contains the information about how these n_{proc} `.vtu` files are set together and which ghost cells they possibly share, in order to together obtain the complete MV geometry of the respective time step \mathbf{ts} . See below, for sample structures of the `.pvtu` file (Listing 5.4), and a `.vtu` file (Listing 5.5) out of a series of $n_{\text{proc}} = 4$ `.vtu` files, which have been computed by 4 different processes, i.e., possibly on 4 compute nodes.

Listing 5.4: Sample content of a `.pvtu` file, which links four different `.vtu` files.

```

1 <VTKFile type="PUnstructuredGrid" version="0.1" byte_order="LittleEndian">
2   <PUnstructuredGrid GhostLevel="0">
3     <PPointData Scalars="Material_Id" />
4     <PCellData Scalars="_Sub_Domain_" />
5     <PPoints>

```

```

6     <PdataArray type="Float64" NumberOfComponents="3" />
7     </PPoints>
8     <Piece Source="MVR_elast_sim_solution_np4_ts.0000_0.vtu" />
9     <Piece Source="MVR_elast_sim_solution_np4_ts.0000_1.vtu" />
10    <Piece Source="MVR_elast_sim_solution_np4_ts.0000_2.vtu" />
11    <Piece Source="MVR_elast_sim_solution_np4_ts.0000_3.vtu" />
12    </PUnstructuredGrid>
13 </VTKFile>

```

Listing 5.5: Sample content of a `.vtu` file, which represents 1 part (out of 4) of the MV geometry.

```

1 <VTKFile type="UnstructuredGrid" version="0.1" byte_order="LittleEndian">
2   <UnstructuredGrid>
3     <Piece NumberOfPoints="3633" NumberOfCells="10777">
4       <CellData ... </CellData>
5       <Points>
6         <DataArray type="Float64" Name="Array" NumberOfComponents="3"
7           format="ascii" RangeMin="0" RangeMax="121">
8           70.0151 77.3317 110.906 69.5631 77.1732 111.169 ...
9         </DataArray>
10        </Points>
11        <Cells>
12          <DataArray type="Int64" Name="connectivity" format="ascii"
13            RangeMin="0" RangeMax="3632">
14            0 1 2 3 4 5 6 7 8 9 10 11 ...
15          </DataArray>
16        </Cells>
17      </Piece>
18    </UnstructuredGrid>
19 </VTKFile>

```

The obtained `.vtu` files (Listing 5.5) contain the information on the coordinates of the deformed nodes of the mesh representation (unstructured grid) of the Mitral Valve, as well as their connectivity matrices. Along with these they allow for the computation of the *von Mises stress* on the leaflet surface, see Section 6.3.

5.2.2 Code Optimization and High-Performance Computing (HPC)

Standard simulation softwares offer a broad variety of methods and tools for setting up simulations in different application areas. However, these simulations are rarely perfectly adapted to a respectively specific model. Knowing that model and simulation type – once adapted for a specific application – often remain the same for large series of runs within a respective area, it is obvious that numerical and performance optimization is usually worth the efforts.

The adaptation of the numerical solution schemes to a model's particular dependency structure allows to considerably accelerate the simulation. For example, coefficient matrices arising inside an algorithm for the discretization of PDEs often exhibit typical patterns (e.g., blocks, sparsity structures) for one model class, such that one can specifically manufacture a storage scheme and a solution technique with respect to the particular matrix and problem type. Numerics and numerical analysis provide the necessary tools to adapt the solution and storage schemes of open-source simulation softwares in order to obtain optimal performance for specific

applications, such as our MVR soft tissue simulation application. Accordingly, through numerical analysis, we intend to customize our numerical algorithms to the given model and simulation class, and to thus obtain significantly improved performance, i.e., e.g., higher accuracy where needed, generally reduced run-times and lower storage requirements.

Using an inhouse-developed FEM software toolkit, HiFlow³, we are able to actually adapt a simulation code in these specific ways, since the HiFlow³ core is open-source, flexibly extensible, completely under (our) control, and all known by the members of the EMCL work group, such that every single line of the code can flexibly be adapted, extended and optimized for specific purposes. Also, the underlying hardware-aware computing concept enables the maximal exploitation of the available computing infrastructure.

This being said, we exemplarily look at different parts of the above presented code structure, which allowed for promising optimization in terms of performance, accuracy and storage requirements. We have outlined some of these in our work [SES⁺15] and will present these in more detail in the following paragraphs.

Two-Level-Parallelism: As the basis for high performance simulations on modern computing systems, parallelism plays the most important role in our HiFlow³-based MVR soft tissue simulation. According to the core structure of HiFlow³, we distinguish two fundamental levels of parallelism: On the global level, we have *coarse-grained parallelism* by means of inter-node distributed grids and correspondingly distributed matrix/vector data structures. On the local level, we have *fine-grained parallelism* by means of platform-optimized linear algebra back-ends.

Through *domain decomposition* using METIS [ARK06], we partition and distribute the MV mesh over several processes, see Figure 5.1 for an illustration. Global MPI-based communication then allows for the exchange of mesh data and of distributed matrix/vector structures between the respective subdomains, i.e., between the employed processes. The *global linear algebra toolbox* provides the tools to deal with the respective parallel matrix and vector structures. For matrix-vector operations, we then exploit the usage of so-called *ghost layers*, which store the needed vector entries for the nodes on the interface between subdomains and thus allow to minimize the domain decomposition-induced communication overhead. In this context, matrices are divided into two parts: one is representing the couplings within a subdomain (diagonal elements), and the other one the couplings between different subdomains (off-diagonal elements). To gain higher flexibility, the HiFlow³ global LA toolbox dynamically uses the local multi-platform structures on each subdomain. Lastly, the thus given coarse-grained parallelism is also reflected in the parallel mesh I/O in `vtk` format (see Section 5.2.1), which enables the efficient simulation input and output handling, e.g., for visualization purposes.

Opposed to this, on the local level, we have *platform-optimized linear algebra back-ends*, and thus allow for highly-efficient parallel on-node routines. Making use of the *local multi-platform Linear Algebra toolbox* (`ImpLAToolbox`) of HiFlow³, which is a C++ toolbox for vectors and sparse matrices that provides interfaces to Intel and AMD multi-core CPUs and to NVIDIA GPUs, we are capable of exploiting *fine-grained and hybrid parallelism*.

For example, using the OpenMP parallelization concepts, we have parallelized the loop structures in our assembly routines, as mentioned in Section 5.2.1. For more details, we refer to our source code along with the given comments and explanations.

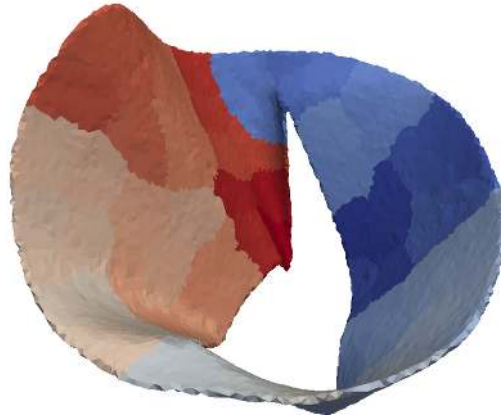


Figure 5.1: Illustration of a partitioned MV mesh representation as used for a HiFlow³-based MVR soft tissue simulation. The decomposed domain is made up from 64 subdomains and was obtained via the HiFlow³ interface to METIS [ARK06].

Also, we have implemented an interface between our MVR simulation code and the HiFlow³-compatible CUDA implementation of the CG solver [AAB⁺10] along with the correspondingly needed data structures, in order to allow for the usage of the computing power of general-purpose GPUs (GPUs) and (GP)GPU clusters. Especially for tasks where large blocks of data are processed in parallel, e.g. arising from the assembly or from the solving part of FE algorithms, due to the highly parallel structure of (GP)GPUs, these are more efficient than general-purpose CPUs.

However, we cannot yet fully exploit the potential of GPUs with our code, since HiFlow³ does not have a CUDA implementation of a preconditioner yet. Thus, at the current state, the necessary forward/backward memory transfer between host (CPU) and device (GPU), which is needed in every time step, consumes more time than what the speedup of the GPU-based solver (without preconditioner) would save.

Thus, summarizing, we have set-up our MVR soft tissue simulation application such that it is compatible with and maximally exploits the HiFlow³ modules and core implementation. Adopting a two-pronged approach, we have parallelized our code both on the global and the local level, and specifically point out the assembly and solver part as well as the respective data structures, which are essential for maintaining scalability and providing high efficiency.

Parallelization of contact search: Accounting for contact in an instationary simulation first of all requires the search for potential contact occurrences in every time step. This treatment of contact is a highly critical and very costly task, and the algorithms for detecting contact are of utmost importance, especially when contact can occur at non-predictable surfaces.

The number of operations needed to check $\mathcal{O}(N)$ surfaces of a discretized domain for potential contact is $\mathcal{O}(N^2)$, such that carrying out these checks in every time step easily lets the search dominate the overall time of the respective step in the loop and even of the entire computation. The contact analysis hence needs to be designed in the most efficient way.

Therefore, we edit the boundary mesh in a cheap preprocessing step, see Section 6.2.1, in a

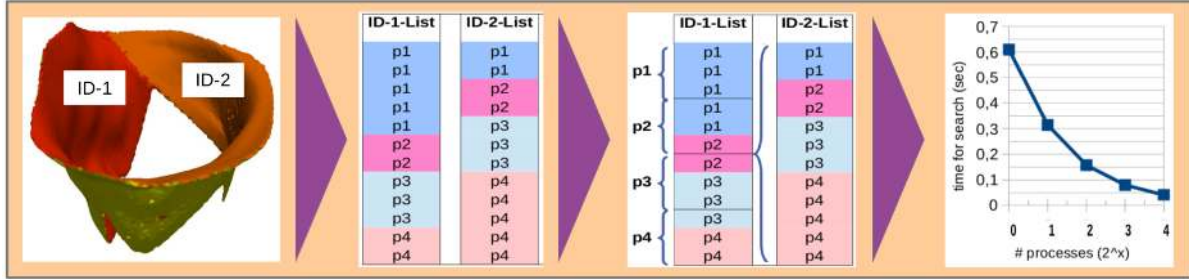


Figure 5.2: Illustration of the contact search parallelization setup by means of a double redistribution of point-ID-lists in our MVR soft tissue simulation. Part 1 shows the ID distribution on the MV surface. Part 2 shows, by the example of 4 parallel processes p1 to p4, how the mesh is randomly partitioned and how the ID lists thus are unequally distributed over the 4 (color-coded) processes. Part 3 shows how our dedicated algorithm equally re-distributes the list pairs to the available number of processes. Finally, part 4 presents a scalability study, which demonstrates how for an increasing number of parallel processes the time needed for the contact search drastically decreases. This indicates the importance and reasonability of our contact search implementation and of the underlying data structures.

way such that, firstly, the two leaflets are assigned with two different IDs, and secondly, the upper and lower sides of the leaflets can be distinguished, too; see Figure 5.2 (part 1). Assuming that either the upper side of anterior leaflet gets into contact with the upper side of the posterior leaflet, or vice versa, and thus excluding the possibility of contact on the lower sides and between two different leaflet parts of one single leaflet (as motivated in Section 4.2.1.1), with the given IDs, we can directly decrease the amount of operations needed for the contact search to approx. $\frac{1}{16}$ of the original amount of operations needed.

Then, once only before the first time step of the simulation, we compose two lists of respectively the two IDs and of their corresponding facets, see Figure 5.2 (part 2). Initially, the IDs may (depending on the preceding coincidental domain decomposition result) be unequally distributed to the employed processes. However, a dedicated, even re-distribution of the complete list-pair to the available number of processes via MPI, see Figure 5.2 (part 3), then allows us to perform a perfectly parallel search with respect to only the potential contact facets in every time step. The resulting contact search algorithm shows very good scalability properties with only a relatively small communication and synchronization overhead; see Figure 5.2 (part 4). Note that this is highly satisfactory, as the MV mesh is not actually suitable for extreme parallelization, since it is strongly surface-dominated and has a very bad volume-to-surface ratio. Hence, the more we parallelize the worse gets the ratio of process-owned cells to ghost-cells.

Once the contact facets are found, we apply a penalty method to penalize the advent of contact, see again Sections 4.1.4 and 4.2.1.1. The penalty term, which counteracts the approach and interpenetration of the two leaflets, is scaled with respect to the absolute value of the gap distance d_{gap} and with respect to the velocity of its decrease over three subsequent time steps.

Numerical Problem Analysis and Efficient Solver/Preconditioner Combinations: Investigating on the analytical and numerical properties of our contact elasticity problem formulation (eq. 4.113) along with the chosen Newmark time integration scheme, and looking at the therefrom arising linear system which must be assembled in every time step, we find that we are dealing with a symmetric, positive definite, sparse system matrix.

When it comes to the choice of a suitable solver and preconditioner, these properties can be considered, and for performance improvement, they should preferably be accordingly reflected. Dealing with symmetric, positive definite matrices, an obviously good choice will be given by the *Conjugate Gradient* (CG) solver [Saa03], possibly along with the *Symmetric Gauss-Seidel* preconditioner [Mei15].

It is to be remarked that the simulation algorithm's efficiency will degrade with increasing number of processors. This is due to two aspects: First, for solving linear systems, communication and synchronization is necessary in every iteration of the CG solver since the systems are fully coupled. The computational domain is divided into the same number of subdomains as the number of processors used. Hence, assuming a constant problem size, the computation to communication ratio decreases when using more processors. Second, the application of a block-Jacobi preconditioner (such as the Symmetric block-Gauss-Seidel preconditioner) is sub-optimal, since the couplings from different subdomains are neglected in the preconditioning step. Therefore, the preconditioner becomes less effective the more subdomains one has to deal with, since more and more information about the inter-domain couplings between the unknowns is ignored. The CG solver will thus need to perform more iterations to achieve the desired stopping tolerance, and hence cause increased computation times.

As the natural consequence, one may easily figure that a multi-level-based preconditioner (available, e.g., in the PETSc [BBE⁺08] or Hypre [FJY06] libraries) would overcome this problem, and hence allow for better performance. Thus, for large systems, we make use of the CG solver along with the *Boomer AMG* (Algebraic MultiGrid) preconditioner from Hypre, for which there is an interface to HiFlow³.

Lastly, it should be noted that there are several parameters with respect to the solver/preconditioner performance, which can be calibrated according to the given problem or use case; most importantly, the number of iterations of the linear solver, and tolerance threshold values. Among others, the relative tolerance value (10^{-x}) ensures that the residual norm of the CG iteration decreases by x orders of magnitude compared to the residual of the starting solution. Besides that, there are lots of fine tuning options for the Boomer AMG preconditioner from Hypre, for which we refer to the dedicated literature [FJY06].

Scenario Management: As explained above, an increased number of parallel processes for a specific MVR simulation scenario is only reasonable up to a certain threshold, as from which point on – for an assumingly constant problem size – the computation to communication ratio decreases and efficiency degrades. However, since we are dealing with lots of scenario-intrinsic uncertainties, we may additionally make use of naturally given options for parallelization. Namely, we also distribute the patient-specifically relevant simulation scenario variations to the available compute infrastructure.

This means, for example, when dealing with a patient, whose tissue properties are not precisely known, and for whom two different types and sizes of annuloplasty ring prostheses

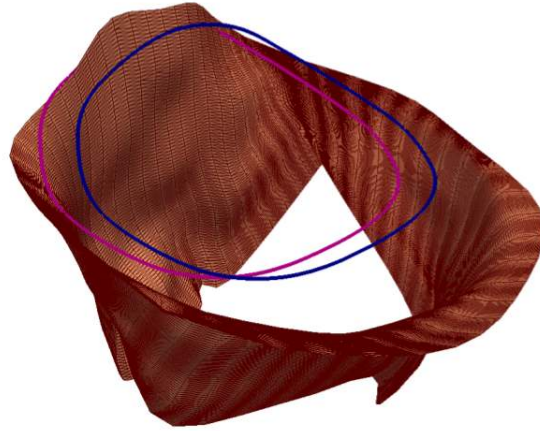


Figure 5.3: Illustration of a possible MVR simulation scenario parallelization: Two different rings may be suitable for a patient’s given MV morphology.

may potentially be suitable, then we might set up a set of six MVR simulation scenarios: two different rings combined with, respectively, three different material parameter sets. See Figure 5.3 for a sample illustration of a scenario parallelization for different ring prostheses.

The thus obtained MVR simulation scenarios are naturally independent, and may hence be arbitrarily distributed to the available HPC infrastructure. Beyond that, one again further parallelizes as described in the preceding paragraphs above.

5.2.3 Outlook and Transition to the next two Chapters

In the previous two subsections, in 5.2.1, we have focussed on the specific structure and setup of our FEM-based MVR simulation application and on the simulation input and control interfaces, and in 5.2.2, we have presented ways in which we optimized our simulation code with respect to performance and accuracy.

As has been motivated above, in order for biomechanical simulations to be really meaningful to the surgeon in the OR, they not only need to perform in almost real-time and deliver highly accurate simulation results, but also need to account for patient-specific medical data and integrate surgical expert knowledge in the model and simulation setup.

In the next two chapters 6 and 7, we will describe how – based on the above-presented simulation code setup and on its control mechanisms and interfaces – we facilitate patient-specific and surgical expert knowledge-based MVR simulations.

We will see that *patient-specificity* and *cognition-guidance* is achieved through an intelligent composition of the `.xml` HiFlow³ simulation scenario input file (as structurally presented in Listing 5.3). This simulation scenario input file describes the entire simulation scenario setup, and thus acts as our simulation scenario control interface. It will be composed by means of a comprehensive simulation preprocessing pipeline which in turn is integrated into a cognitive data/information processing system.

In Chapter 6, we will give detailed information on the entire biomechanical modeling and simulation workflow and the general simulation scenario management. Our focus will be put on our simulation pre- and post-processing pipelines for MVR surgery assistance, which are comprehensive, patient-specific and fully-automated. In Chapter 7, we supplementally look at the underlying integrative, cognitive data and knowledge architecture, which allows for the overall cognition-guidance, and which is to facilitate the combination of all steps in the MVR surgery assistance workflow.

We conclude with the remark that our simulation application represents a *prototypic work* only, and as such does not claim for perfection on its own. Rather, it is supposed to hold as a *research prototype* in order to allow for investigating on the integration of the simulation into a *simulation-based, patient-specific, cognitive surgery assistance system*.

6 Biomechanical Model and Simulation Management by means of the MSML

Having derived the numerical solution of the MVR contact elasticity problem (see Chapter 4), and – based on these findings – having implemented our FEM-based MVR simulation application along with the respective simulation input and control interfaces (see Chapter 5), in this chapter, we will focus on the setup of adequate, patient-specific MVR simulation scenarios, that can be executed by the simulation application and yield appropriate results for MVR surgery assistance.

Therefore, in Section 6.1, we present the *Medical Simulation Markup Language* (MSML), which is to simplify the complex biomechanical modeling and simulation workflow (from medical imaging via segmentation and meshing to modeling and simulation), and which is to facilitate the patient-specific setup of general biomechanical simulation scenarios.

In Sections 6.2 and 6.3, we will demonstrate the application of the MSML for the clinical use case of MVR surgery. On the one hand, we present a fully automated pipeline of simulation preprocessing operators which – integrated into the MSML – enable the setup of comprehensive, patient-specific MVR surgery simulation scenarios. On the other hand, we propose a set of postprocessing tools which may enhance MVR surgery assistance through the meaningful supply of additional clinically relevant simulation-based information to the surgeon in the OR.

Lastly, in Section 6.4, we will discuss the presented work, particularly in the context of our developments on a simulation-enhanced MVR surgery assistance system, and propose further intended work to extend the MSML in the future.

6.1 The Medical Simulation Markup Language (MSML)

In the preceding two chapters it has been described how we can model and simulate the behavior of the elastic MV tissue on the basis of the elasticity theory. Using the FEM simulation toolkit HiFlow³, and describing the respective MVR simulation scenario by means of the before-mentioned `xml` input file, we can run a simulation and obtain simulation results, which are to be provided to surgeons in order to assist them in their cardiac operation.

However, we have not yet looked at the complex biomechanical modeling workflow, which is to compose these `xml` scenario input files along with the respectively needed specifications, like mesh geometries, material parameters, boundary conditions, etc.

This biomechanical modeling workflow usually covers many different steps, see Figure 6.1, the most important ones of which were presented and described in the context of the state-of-the-art, see again Chapter 3: Commonly, it starts from tomographic data, goes via image segmentation and FE mesh generation, and also includes the definition of boundary conditions, material parameters and other model and simulation specifications; compare Section 3.1.5.2.

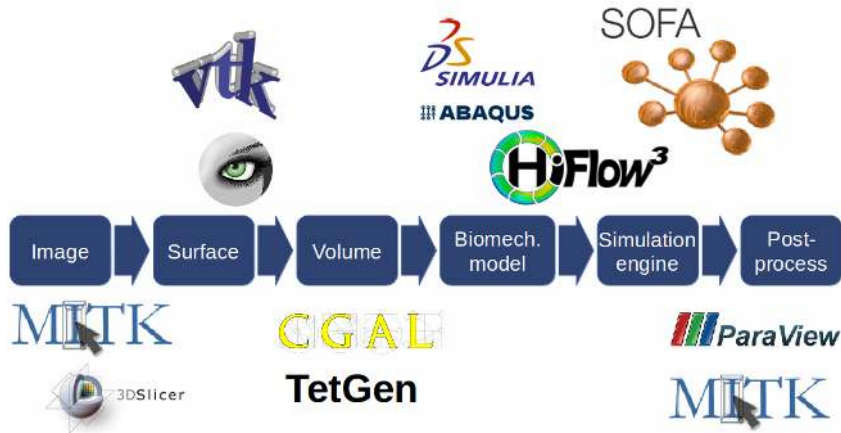


Figure 6.1: Schematic representation of the complex biomechanical modeling workflow from tomographic images via simulation to postprocessing, showing the miscellaneous specific tools and software components used for the different stages.

Even though being a seemingly simple task, the composition of a reasonable biomechanical model as a result of passing through the above-named workflow steps can be very time-consuming in practice, and often requires manual interaction. This is mainly due to the fact that there is a large set of specified software tools, which separately address the miscellaneous tasks in the different workflow steps, see Figure 6.1 and again Section 3.1.5.2. Also, as a consequence, there exist many different data formats, which are rarely directly compatible, and there is a severe lack of standards concerning the interfaces between the respective tools.

With the modeling approach of the *Medical Simulation Markup Language* (MSML) [SSS⁺14], we seek to describe and integrate the complete workflow and its many different steps, and finally aim at facilitating the patient-specific description and construction of biomechanical models from tomographic data.

The MSML scheme therefore describes a biomechanical model and simulation including all relevant information such as mesh geometry, material model, boundary conditions, as well as simulation-specific properties like solver and parallelization, etc. The unified description is parsed and interpreted, and can then be exported to different simulation engines or FE solvers. The unique feature of the MSML is its ability to *not only* describe the biomechanical model and simulation, *but* to also cover all steps in the modeling and simulation workflow, and to thus act as a middleware between all tools that are used in the modeling pipeline.

In the following, based on the joint work [SSS⁺14], we will give details on the MSML modeling scheme and on the MSML workflow description. We will then focus on the Python-based MSML implementation in general and on the MSML HiFlow³ exporter in particular, which enables exporting MSML-based workflows and their respectively set-up simulation scenarios to the HiFlow³ simulation engine. Lastly, in order to highlight the flexibility, robustness and ease-of-use of the approach, we will give some simple soft tissue modeling examples.

The general idea of the MSML modeling scheme: The general idea of the MSML was the design of a flexible format to describe FEM-based medical simulations and their underlying models, as well as the implementation of a middleware for the integration of miscellaneous software components in the complex biomechanical modeling workflow.

Being a joint research and development effort between research institutions in Karlsruhe and Heidelberg (KIT, DKFZ, Heidelberg University), we started our collaboration with a *requirements analysis* in order to figure and classify the most important modeling properties and functional features. Resulting from this initial discussion, we conjointly found that, in the end, the MSML should at least contain or integrate the following items:

- **Data formats:** tomographic data (`dcm`, `vtk`); segmentation data (`stl`, `vtk`); surface mesh data (`vtk`); volume mesh data (`vtk`, `msh`, `inp`); boundary conditions data (`xml`, `vtk`, `inp`); simulation scenario setup data (`sofa.scn`, `hf3.xml`); output data (`vtk`, `hdf5`);
- **Tools and libraries:** MITK; Netgen, Tetgen, CGAL; Ansys, FEBio, Sofa, Abaqus, HiFlow3; ParaView.
- **Mesh types:** surface mesh, volume mesh; structured or unstructured meshes; tetrahedral or hexahedral elements; linear or quadratic elements;
- **Material models:** linear elastic, corotational, hyperelastic, viscoelastic;
- **Boundary conditions:** fixed or displacement (Dirichlet) BCs, pressure (Neumann) BCs, contact BCs.

The MSML alphabet and the model description: Inspired by the above requirements analysis, we have set up an *alphabet* as the foundation of our model description: the *MSML Scheme*, with everyone defining and implementing some of the required features. This alphabet is defined via the `msml.xsd` scheme file, which specifies how `msml.xml` files are classified, how their structure and contents are syntactically described, and how they have to be composed with respect to their elements, values, etc.

Several elements remain the same for all types of biomechanical simulations, regardless if solid mechanics or fluid dynamics problems are considered, and even irrespective of the fact if they are FE-based or not. These so-called *fixed elements* comprise, e.g., the *physicsModel*, which contains the *mesh* (which specifies the type of mesh and its elements), several *materialRegions* (which denote material models for different areas that are specified via *indexGroups* or *IDs*), and a set of *constraintRegions* (which describe the respectively active boundary conditions). The *physicsElement* can hence contain a *contactModel*, too. See Listing 6.1 for an example of a schematic xml-based model description as an excerpt from an `msml.xml` file.

Listing 6.1: Sample excerpt of an xml-based model description.

```
1 <scene>
2   <object id="liver">
3     <mesh>
4       ...
5     </mesh>
6     <material>
7       <region id="liverMaterial">
```

```

8         <linearElasticMaterial youngModulus="4166" poissonRatio="0.488"/>
9         ...
10        </region>
11    </material>
12    <constraints>
13        <constraint name="bodyConstraint" forSimulationStep="\${initial}">
14            <fixedConstraint indices="\${set1}"/>
15            <displacementConstraint indices="\${set2}" displacement="..."/>
16        </constraint>
17        ...
18    </constraints>
19 </object>
20 </scene>

```

Next, the element *environment* contains *solver* and *simulation* specifications. These, e.g., describe global parameters about the solver (i.e., if an ODE or a linear systems solver is used, which hardware and what type of parallelization is employed, etc.). The element *simulation* contains *simulationSteps* which comprise the constraints, external loads and general boundary conditions that can be prescribed for all stages of the simulation separately.

Generally, an *MSML Scene* can thus be divided into several *objects*. Besides the above described elements, all other components of the final MSML alphabet (such as material models or mesh types) are defined by means of a hierarchy of *xml* files. This allows for the easy extensibility of the MSML alphabet and yields its enormous flexibility: New material models or boundary condition types can be defined and integrated into the MSML simply by adding an *xml* file, which lists the respectively contained argument names, parameters, logical/physical types, values, etc.

Finally, the idea behind the unified MSML description is to enable an easy export to different simulation applications or FE solvers by means of so-called export operators (or *exporters*). Such an exporter is an operator that generates native input files for simulation applications like Abaqus, HiFlow³, or Sofa. For instance, referring to HiFlow³, such a native input file is given by an *xml* scenario setup input file, see again Section 5.1.2.

However, not all export targets necessarily support the entire set of elements and functionalities described in the MSML alphabet. For instance, the HiFlow³ core does not contain the functionality to consider contact boundaries, such that – to the MSML – the export target HiFlow³ only supports a subset of the MSML alphabet.

For this reason, every MSML exporter holds a tree of compatible nodes, which allows for an easy compatibility checking scheme that ensures the consistency between the MSML description and the respective FE solver or simulation application.

For more information, we refer to the open-source implementation on GitHub: <http://github.com/CognitionGuidedSurgery/msml/>. The current MSML alphabet can be found via: `msml` → `share` → `alphabet`.

The MSML workflow description: As motivated at the beginning of this Section, the unique feature of the MSML is, that it not only describes the biomechanical model and simulation scenario as explained just above, but that it also covers the entire modeling workflow (i.e., how the respective model is obtained), and that it acts as a middleware between the different steps in this workflow.

When looking at the different steps of the modeling workflow, each step is associated with certain data types, e.g., images, segmentations, meshes, or indices lists. To connect these, the MSML allows to link the respective data types through so-called *operators*. See Listing 6.2 for a simple workflow model as described as part of an `msml.xml` file.

Listing 6.2: Sample xml-based workflow model to produce an FE mesh from a segmentation.

```

1 <workflow>
2   <CGALmeshGeneratorI2V id="liverVolumeMesher" image="\${voxelizer}"
3     meshFilename="liverSurfaceFromVoxelImage.vtk" />
4   <VTKvoxelizer id="voxelizer" surfaceMesh="\${vtkConverter}"
5     imageFilename="liverVoxelImage.vti" resolution="100" />
6   <stl2vtk id="vtkConverter" STLMesh="Liver.stl"
7     vtkFilename="liverSurface.vtk" />
8 </workflow>

```

Generally, we distinguish two types of operators: One the one hand, they can be defined to link data types from two different stages of the workflow; for example, a segmentation operator links tomographic data to surface meshes, or a mesh generation operator links surface meshes to volume meshes. On the other hand, they can link data that belong to one stage of the workflow, such as a surface mesh smoother or a file format converter. Again, an xml file defines for each operator the input and output data types, the data formats, the set of operator parameters and possibly ancillary information on how the operator can be executed; see Listing 6.3 for an example. As a prerequisite, the respective data format and data type descriptions necessarily need to be defined in the MSML alphabet.

Listing 6.3: Sample xml-based operator description of a *Marching Cubes* operator.

```

1 <msml:operator name="MarchingCubesOperator">
2   <input>
3     <image ... />
4   </input>
5   <output>
6     <surfaceMesh ... />
7   </output>
8   <parameters>
9     <voxelSize ... />
10    ...
11  </parameters>
12 </msml:operator>

```

As a result, this ensures *type safety* and enables *compatibility checking* between operators and their input and output elements. If incompatible data formats are detected, but the same data type is used, the system can automatically use the list of known operators to suggest an appropriate *conversion operator*. Due to the fact that any arbitrary existing tool (which may be available as an executable or via a library) can be integrated into the MSML by simply adding an xml file, it becomes very easy to include external established software tools into the MSML scheme. As such, the MSML hence acts as a *middleware* between all tools used in the biomechanical modeling pipeline.

For details and exemplary operators, we refer to the open-source implementation on GitHub: <http://github.com/CognitionGuidedSurgery/msml/>. A list of all currently available operators can be found via: `msml` → `share` → `alphabet` → `msml_operators`.

The MSML implementation: The MSML is implemented as a *hybrid* Python/C++ library. All *high level functions* such as `xml`-parsing, compatibility checking, and operator calls are performed by the Python part. Above all, a Python module (`xml.py`) analyzes and interprets as an `xml`-parser the `xml`-based MSML biomechanical model description and the `xml`-based MSML workflow, and transfers the valid `msml.xml` files into the Python-based MSML-internal scenario data structure. Consequently, the Python part simultaneously performs a validation of the modeling workflow and checks the consistency of the workflow components based on the MSML `xml`-scheme.

Also, the MSML bridges through *Python bindings* to a set of C++ operator libraries like CGAL, VTK, and Tetgen. These operators can either be executed as a function call or as a command line tool. Again, please note that this allows for a flexible integration of many different third party tools. For instance, the current operator collection includes different operators based on CGAL, Gmsh, Tetgen, VCGLib and the Visualization Toolkit (VTK). These operators – integrated and linked as required – provide miscellaneous functionalities, such as general surface mesh generation, surface mesh size reduction, linear/quadratic tetrahedral volume mesh generation, mesh quality improvements, extraction of point and element indices based on different criteria, summarizing elements, points or surface facets to sets and regions, as well as post processing operations, such as mesh coloring, mesh comparison, and mesh threshold highlighting. For performance reasons, most of these operators are implemented in C++ and wrapped via Python bindings such that they are accessible by the MSML library.

With respect to the *data handling*, it should be mentioned that, while it is, of course, possible to store all data associated with an MSML model into a single `xml` file, it is often more computationally efficient to leave large image and mesh data in their native formats and to just reference this data via filenames. Therefore, all referenced data is copied to a single folder, which can then be compressed to reduce the storage requirements. In the current implementation, we use the `vtk` data types originating from the Visualization Toolkit (VTK), e.g., `vti` for images and `(p)vtu` or `vtk` for meshes. Yet, of course, it is possible to use different additional formats thanks to the flexibility and extensibility of the MSML alphabet.

Summarizing, via its Python-based implementation, the MSML allows to relate data and environments (that are defined in the MSML model description), and to check compatibilities, dependencies and parameter options with respect to the alphabet by going through its hierarchical structure. Furthermore, it facilitates the biomechanical modeling workflow through the (automated) execution of operators.

The MSML HiFlow³ exporter: Once the MSML model description, as defined in an `msml.xml` file, has been transferred into the Python-based MSML data structure, an *export operator* (or *exporter*) is to translate and transform this data such that the emerging files can be used as input by different simulation engines. Here, we particularly look at the MSML HiFlow³ exporter, which generates native HiFlow³ simulation input files for a HiFlow³ simulation application.

As has been explained in the previous chapter, in Section 5.1.2, a HiFlow³-based simulation application requires as input an `.xml` scenario description, which contains, among others, a list of important problem- and simulation-related parameters, and also a link to a mesh and a set of one or more boundary condition input files.

Hence, derived from the biomechanical MSML model and scene description, and thus based

on the resulting scenario-specific MSML-internal scenario data structure, the task of the HiFlow³ exporter is to process the scenario-specific MSML elements, values, parameters and specifications into

- a `hf3.xml` HiFlow³ simulation scenario input file, which additionally links
- a `mesh.vtu` or `mesh.inp` mesh morphology file, and
- a `bcdata.xml` boundary conditions specification file,

in order to thus fully define the respective HiFlow³ simulation scenario; see again Section 5.2.1.

Eventually, the HiFlow³ exporter sends the command to execute the simulation to the operating system: It calls the HiFlow³ simulation application with the respectively passed `hf3.xml` simulation scenario input file, and executes it on respectively n parallel processes.

It has to be emphasized, that the HiFlow³-based simulation scenario is hence *fully controllable* by means of the MSML and through a scenario-specific MSML model description, which simultaneously facilitates the respectively preceding model setup workflow from image to biomechanical model.

For details on the implementation of the MSML HiFlow³ exporter, we refer to GitHub <http://github.com/CognitionGuidedSurgery/msml/> and lead through to: `msml` → `src` → `msml` → `exporter` → `hiflow3`.

Going further, in the next Section 6.2, we will show how we thus achieve *patient-specificity* and the overall *intelligent composition* of our MVR simulations: namely, through the MSML and its HiFlow³ Mitral exporter, that creates the input files for our HiFlow³-based MVR simulation application, which in turn reads in these input files by means of the HiFlow³ `xml` input interface.

At this point, we remark that the above described *HiFlow³ exporter* is designed for the setup of general (non-specific) biomechanical FEM simulations. Opposed to this, below in Section 6.2, we will present the *HiFlow³ Mitral exporter*, which is a derived class and inherits from the HiFlow³ exporter. The HiFlow³ Mitral exporter is designed for the specific purpose of MVR simulations that are to be run using our specific and purpose-optimized HiFlow³-based MVR simulation application, which was described in Section 5.2.

Example MSML models and workflows: In order to highlight the flexibility, robustness and ease-of-use of the MSML, we will give some simple soft tissue modeling examples in the following. Beyond the below examples, we again refer to the open-source implementation of the MSML that is accessible via GitHub: <http://github.com/CognitionGuidedSurgery/msml/>. Here, a list of examples is provided, too, and can be found via: `msml` → `examples`. Additionally, we refer to the next Section 6.2, where we give detailed explanations on how we adapted, extended and employed the MSML for the specific MVR use case.

Example 1: Using the MSML to set up an HPC-based HiFlow³ soft tissue simulation:

The first example refers to the `msml.xml` file which is listed on GitHub under https://github.com/CognitionGuidedSurgery/msml/blob/experimental/examples/BunnyExample/bunnyCGAL_HiFlow3.msml.xml.

The MSML was employed to set up and run a biomechanical soft tissue simulation of the deformation of the *Stanford Bunny* [TL94], where the open-source multi-purpose finite element software HiFlow³ was employed as simulation engine. Having defined the workflow to derive

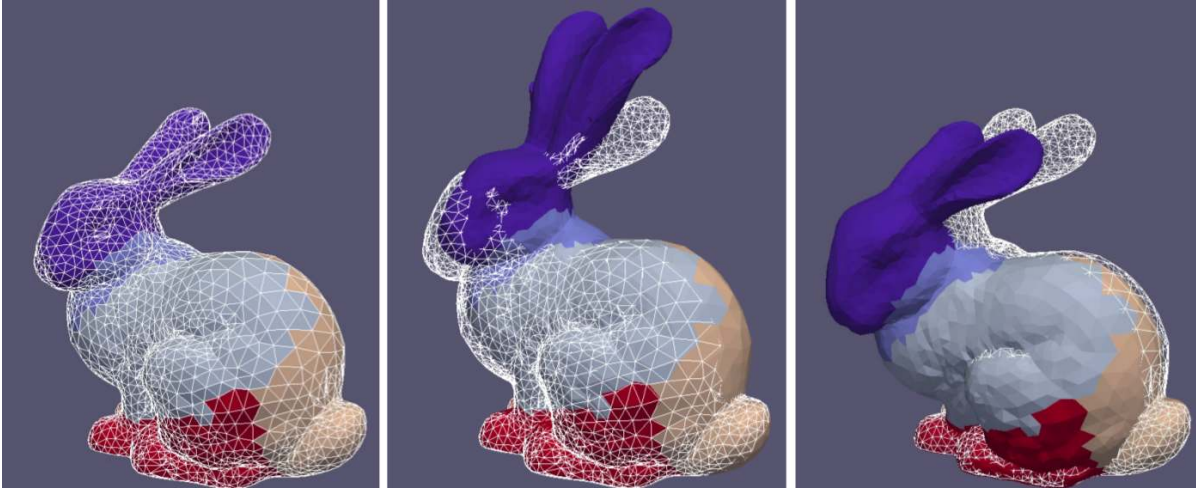


Figure 6.2: Screenshots from the HiFlow³-based soft tissue simulation scenario with the Stanford Bunny. The left picture shows the initial state and hence the result of the MSML-based FE volume mesh generator. The middle and right pictures show states resulting from the simulation scenario steps as given in the `msml.xml` file.

an FE volume mesh from the bunny’s given surface mesh, and after then declaring the bunny’s body region, different *indexGroups*, as well as *constraintRegions* for a chain of subsequent *simulationSteps*, we have been able to run the simulation using HiFlow³. We exemplarily parallelized the simulation to five parallel processes and made use of GPUs for the actual solving step in the simulation to indicate a simple High-Performance Computing testing scenario which aims at real-time simulation. The access and control of the model parameters (like geometry, boundary conditions, material model, tissue properties, etc.) as well as of the simulation settings (like solver-preconditioner combination, degree of parallelization, CPU- or GPU-based computation, etc.) is handled via the said `msml.xml` file. We run a dynamic simulation and use Rayleigh Damping and the Newmark time integration scheme. A symmetric Gauss-Seidel preconditioner is used in combination with an iterative CG solver to solve the resulting linear systems in each time step. See Figure 6.2 for screenshots obtained from the respectively resulting simulation scenario steps.

Example 2: Using the MSML to run the same soft tissue simulation with SOFA.

The second example refers to the `msml.xml` file which is listed on GitHub under <https://github.com/CognitionGuidedSurgery/msml/blob/experimental/examples/BunnyExample/bunny.msml.xml>.

We employed the MSML to set up and run another soft tissue simulation of the deformation of the *Stanford Bunny* [TL94]. Generally corresponding to the first simulation step of example 1, here, instead of HiFlow³, we use the simulation engine SOFA [FDD⁺12], for which an exporter has been implemented by our colleagues at the KIT [SSS⁺14]. Again, we run a dynamic simulation, yet, this time we employ the Implicit Euler time integration scheme. Using an iterative CG solver, the linear systems of each time step are solved on CPUs instead of GPUs.

Please note and that the MSML hence not only facilitates the biomechanical modeling workflow, and that it not only simplifies the generation of biomechanical models and simulation scenarios either. In fact, the MSML can also export the same simulation scenario to different simulation engines and thus allows for comparing the simulation results. It can hence be employed for *comparison* and *validation* of simulation algorithms and of algorithm specifications, e.g., SOFA vs. HiFlow³, or Newmark vs. Implicit Euler, or CG vs. preconditioned CG, etc.

For more information, we refer to [SSS⁺14] and to the open-source implementation on GitHub: <http://github.com/CognitionGuidedSurgery/msml/>.

Summarizing the features of the MSML: The Medical Simulation Markup Language (MSML) is a flexible, unified xml-based description for the entire biomechanical modeling workflow and for FE-based biomechanical models and simulation scenarios. As a unique feature, it can also act as a middleware between all tools used in the biomechanical modeling pipeline, and thus facilitates and simplifies the modeling process, from tomographic image data via FE meshes to simulation and postprocessing.

The main library is implemented in Python, but it also provides a large collection of C++ operators to facilitate efficient and flexible biomechanical modeling workflows. The xml-based MSML alphabet is easily extensible and enables a robust compatibility checking. With the HiFlow³ exporter, the MSML allows for biomechanical models and simulation scenarios to be exported to the HiFlow³ simulation engine.

Additional information can be found in our joint work [SSS⁺14]. Also, we refer to the online documentation at <http://msml.readthedocs.io/en/latest/> and to the open-source implementation on GitHub: <http://github.com/CognitionGuidedSurgery/msml/>.

6.2 Comprehensive Simulation Preprocessing for MVR

Having presented with the MSML an approach to describe the general biomechanical modeling workflow – from tomographic images to model and simulation –, we will now have a look at the specific use case scenario of a mitral valve reconstruction (MVR) again.

Our goal is to obtain a pipeline of operators that intelligently analyze and process the patient-specifically given medical data (both image and parameter data), and to thereon-based allow for the setup of *comprehensive, situation-adapted, patient-individual MVR simulation scenarios*. Via integration of these operators into the framework of the MSML, we want to guarantee for the internal compatibility of all data and enable the full automation of the complete MVR simulation preprocessing workflow.

To the best of the author’s knowledge, there has not been such an approach to foster the usage of patient-specific MVR simulations in the OR until 2016, when Schoch et al. [SKS⁺16] first presented their concept and implementation: The proposed system correspondingly processes all available knowledge and information via a fully automated chain of dedicated MVR simulation preprocessing operators, and then sets up MVR simulations that are both comprehensive (with respect to the underlying biomechanical model, as described in Section 4.2) and also patient-specific (with respect to the patients’ image and parameter data).

Figure 6.3 illustrates the idea of our MVR simulation preprocessing workflow: the *red-framed central block* is to process patient-individual image and parameter data (from the *left*

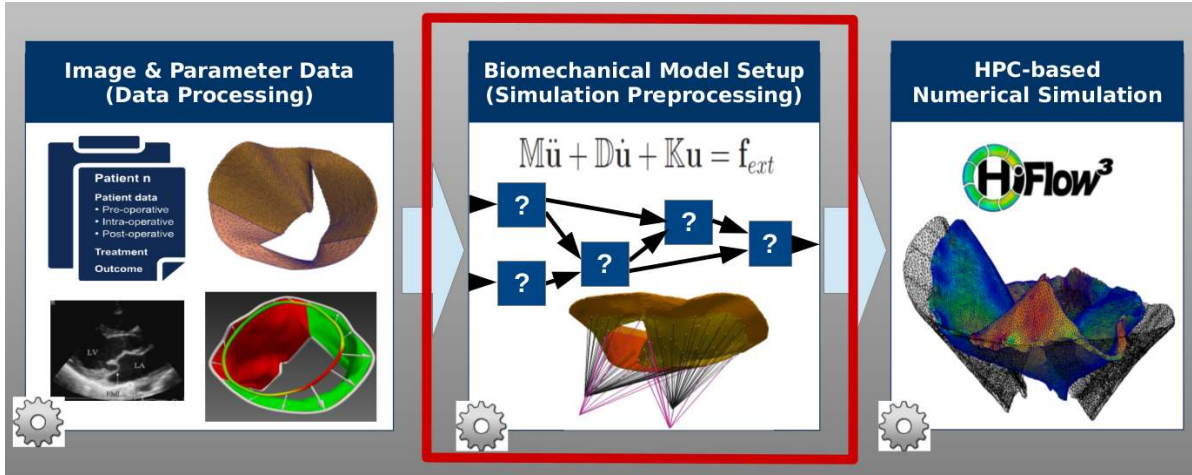


Figure 6.3: Illustration of the simulation-based MVR surgery assistance workflow. The red-framed block (simulation preprocessing) represents the MSML-based chain of MVR information processing operators, which we are dealing with in this Section 6.2. The operator chain processes patient-individual image and parameter data from the preceding block, and is to yield as a result patient-individual, comprehensive MVR simulation scenarios that can directly be executed using the HiFlow³-based MVR simulation application in the subsequent block.

block) via a set of dedicated data analytics and data processing operators in order to thus fully automatically set up comprehensive, patient-specific MVR simulation scenarios, that can directly be executed by means of our HiFlow³-based MVR simulation application, which was presented in Chapter 5 (*right block*).

In the current, prototypic setup of our cardiac surgery assistance system, the input of the MVR simulation preprocessing block is twofold: On the one hand, it consists of a *segmentation*, i.e., a surface representation, of a patient’s MV. On the other hand, it consists of a digitalized model of an patient-specifically suitable *annuloplasty ring prosthesis*.

Both the patient-individual segmentation and the patient-individually fitted annuloplasty ring prosthesis are obtained and made available to our MSML-based chain of MVR information processing operators as part of a complex *system of cognitive tools for MVR surgery assistance* [SPW⁺16] and through the underlying *cognitive data infrastructure and software architecture*, see [SPW⁺16] and [FMK⁺16]. Our work with respect to the overall cognitive system setup will be presented and explained in detail in Chapter 7.

Based on the input data, the biomechanical MVR soft tissue simulation (as in Section 4.2.1) shall accordingly be set-up by means of composing the corresponding patient-specific MVR simulation scenario, i.e., an `hf3.xml` scenario input file along with respective links to geometry and boundary condition data, as defined in Section 5.2.1.

Corresponding to what we have presented for a general biomechanical soft tissue simulation in Section 6.1, this includes for the specific case of MVR surgery, among others, the generation of appropriate geometry input data (i.e., a dedicated FE mesh), and the definition of material

properties and boundary conditions according to the biomechanical model of MVR surgery. Finally, based on the thus declared patient-individual simulation scenario, the HiFlow³-based MVR simulation application (as in Chapter 5) can directly be executed in order to virtually simulate the ring implantation procedure and the patient-specific post-operation behavior of the MV.

6.2.1 Setting up a Chain of Operators for MVR Simulation Preprocessing

In Section 4.2.1, we have motivated and detailed the biomechanical model for the MV and for MVR annuloplasty surgery, and the corresponding BC data structures have been introduced in Section 5.2.1 (in paragraph “The Simulation Scenario Input and Control Interface”). In the following, we will describe a *series of MVR simulation preprocessing operators* that are to facilitate the setup of MVR simulation scenarios through deriving the corresponding data for our model (i.e., mesh, material parameters, and boundary conditions) from the above-mentioned input (i.e., the patient’s MV segmentation and a suitable ring prosthesis model).

Before going into details on the operators themselves, we will have a closer look at the *input data* again, which the operators will need to process. As mentioned above, to the one part, the input for our MVR simulation preprocessing block is composed by *MV segmentations*. These were obtained from 4D transesophageal ultrasound images of the valvular apparatus, on which a semiautomatic segmentation algorithm [ELAM⁺15] had been carried out. For each patient, out of a certain number of 3D images across the entire cardiac cycle, one time step is chosen in which the MV is maximally open and where the two leaflets are clearly separated from each other. The segmented MV geometry of this time step is then represented as a polygonal grid that is constituted of 2D triangular cells. The vertices of this grid are marked with IDs that indicate whether the respective vertex belongs to the anterior or posterior leaflet, or whether it lies on the annulus. On the annulus, there are in total 16 pre-defined suture points with unique IDs (with numbers from 1 to 16). These IDs are given in consecutive order and in counterclockwise direction around the annulus, starting from the anterior commissure point. See Figure 6.4 for an illustration. The average dimensions of the original exported 3D ultrasound image files were 233.6 x 217.6 x 214.4 voxels (with a 0.81 x 0.8 x 0.74 mm³ voxel size).

To the second part, the input consists of a digitalized model of the patient-individually selected *annuloplasty ring prosthesis*. The respective ring has been identified to be (potentially) suitable for the respective patient by means of our deductive system for MVR ring selection [SPW⁺16]. This deductive system is part of a preceding cardiac surgery assistance step, and its underlying reasoning algorithms interpret the patient-individually given medical data on the basis of surgically motivated rules in order to thus determine potentially suitable rings. For more information, we refer to Chapter 7 and to our work [SPW⁺16]. Going further, digitalized models of the respective ring prostheses are obtained by means of algorithms described by Graser et al. [GWAM⁺14]. Every such ring model is also represented by means of a polygonal grid. According to the above-mentioned 16 labeled suture points on the segmentation of the natural MV annulus, the model of the artificial ring prosthesis also contains 16 unique suture points that carry the same IDs. During the ring implantation process, respectively ID-matching points on natural annulus and annuloplasty ring are sutured and thus torn towards each other,

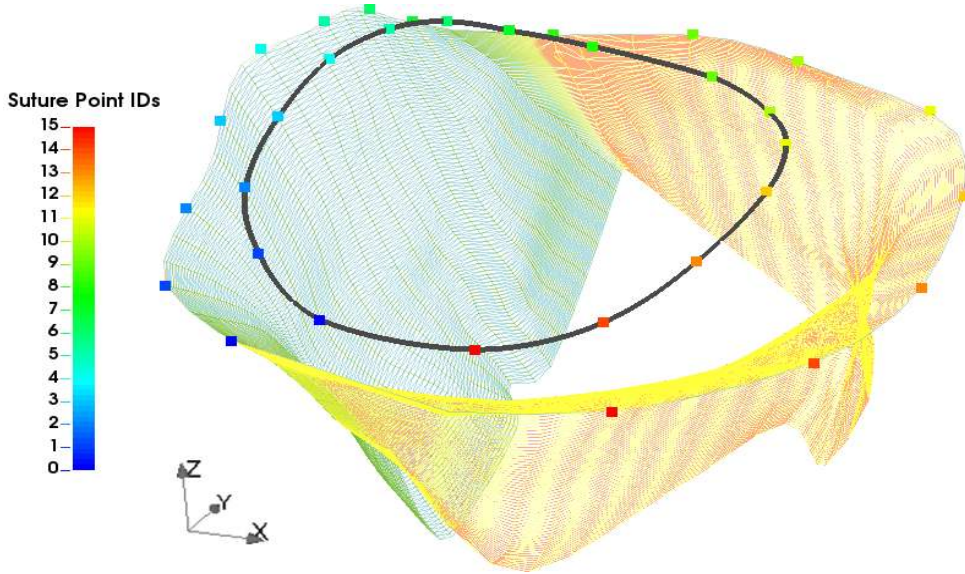


Figure 6.4: Color-coded representation of the vertex IDs on segmented MV geometry and annuloplasty ring prosthesis. The figure shows the anterior (green) and posterior (orange) leaflets, and the described set of pairs of suture points on annulus and annuloplasty ring. These points are again identified by color, and their anatomic names and meanings are given in Table 6.1.

Table 6.1: List of annulus point IDs along with their anatomic names, as provided by [ELAM⁺15], and as motivated and described in [EdSAM⁺16] and [GWAM⁺14].

0	Anterolateral commissure	8	Posteromedial commissure
1 - 3	Control point	9 - 11	Control point
4	Saddle-horn	12	Posterior annulus
5 - 7	Control point	13 - 15	Control point

see again Figures 2.7 and 4.9. As described in Engelhardt et al. [EdSAM⁺16], these 16 control points indicate important landmarks on the ring, such as the saddle horn on the anterior annulus, the point opposite to it on the posterior annulus, or the two commissure points.

Figure 6.4 illustrates a color-coded representation of the segmented MV geometry and a set of pairs of suture points on the natural annulus and respectively on the artificial annuloplasty ring, as obtained from [ELAM⁺15]. Supplementally, Table 6.1 lists the anatomic names of the indicated suture points.

In the following, we will denote the segmented MV geometry by S' and the representation of the annuloplasty ring prosthesis by R .

Based on this input, we start our MVR simulation preprocessing workflow with the application of three *general-purpose operators*, that are already available as part of the MSML for generic usage in arbitrary biomechanical modeling workflows (see again Section 6.1). First, the `SurfaceToVoxelData` operator is applied to convert the surface segmentation S' into a

3D voxel image. Second, the `MarchingCubes` operator further transforms this 3D voxel image into a 3D surface mesh S , and third, the CGAL-based `SurfaceToVolumeMesh` operator finally produces a 3D volume mesh, which we denote by V . The mesh V is an unstructured grid consisting of linear tetrahedra. Its resolution and quality correlate with the initial resolution of the ultrasound image, and it is suitable for FEM-based soft tissue simulation.

Thus, having applied these three general purpose MSML operators, we have transformed the segmented MV geometry into an *FE-compatible mesh* V , the surface of which we call S .

Related to the FE mesh, our MV/MVR model requires the definition of adequate *material properties* and *material regions*. Recalling what has been explained in Section 4.2.1, we assume the MV tissue properties as suggested in the works of Mansi et al. [MVG⁺12] and of Pouch et al. [PXY⁺12]:

- Tissue density: $\rho = 1.04 \frac{\text{g}}{\text{mL}}$.
- Lamé’s first parameter λ in a range of 28 466 Pa and 68 640 Pa, and
- Lamé’s second parameter or shear modulus μ in a range between 700 Pa and 1 400 Pa.

Apart from minor deviations (which would require manual calibration), we apply an average parameter combination of $\lambda = 56\,933$ Pa and $\mu = 1\,200$ Pa, which yields realistic results for the major part of simulated patient MVs.

Lastly, besides mesh and material parameters, our biomechanical MV/MVR model requires for the definition of a *set of specific boundary constraints* (see again Section 4.2.1). In our simulation preprocessing pipeline, we hence need to facilitate the computation of corresponding BC data structures (as listed in Section 5.2.1, paragraph “The Simulation Scenario Input and Control Interface”):

- Facetwise Neumann BCs, to represent the blood pressure pushing onto the leaflets,
- Facetwise Penalty Contact BCs, to account for the event of contact between the leaflets,
- Pointwise Neumann BCs, to imitate the effects of the chordae, and
- Pointwise Dirichlet BCs, to cover the annuloplasty ring implantation process.

Thus, building up on the above three general-purpose MSML operators, in the following, we will present our *pipeline of dedicated MVR simulation preprocessing operators*, which allow to comprehensively analyze patient-specific MVs in order to then set up comprehensive, patient-specific MVR simulation scenarios fully-automatically. In particular, we will go through the process of how the above BCs and their corresponding BC data structures are computed by the respective operators in this pipeline. Please note, that we have described and presented the whole setup in our work [SKS⁺16], and gave additional details on the implementation of the underlying algorithms in [SKS⁺15].

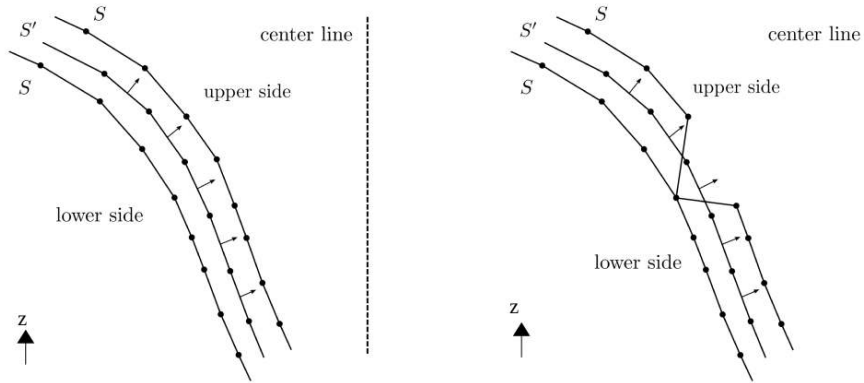


Figure 6.5: Cross-sections of the segmented mitral valve leaflet geometry S' and the surface S of the volume mesh. The respective normal vector fields are shown by means of the arrows. (a) Ideal world case, where the segmented MV geometry S' is perfectly nested inside the volume mesh V . (b) Exception case, where the preceding volume meshing operator has produced a too coarse mesh, which affects that the segmented surface can get outside the nesting volume mesh.

6.2.1.1 Facetwise Neumann BCs (Blood Pressure): Identification of Upper and Lower Side of the Leaflets

Here, we describe a method for the classification of the cells on the MV surface S into cells on the upper and lower side of the MV leaflets. Thereby, the *upper* side of the MV surface is the side on which the blood pushes during diastole, and accordingly, the *lower* side of the leaflets is the side on which it pushes during systole, thereby causing the closure of the MV.

To proceed, we make use of the normal vector field N' of segmented MV geometry S' . This normal vector field is oriented towards the center line of the left ventricle. We refer back to Figure 4.6 for a schematic draft of the ventricle morphology along with the notation used, and to Figure 6.5(a) for a sketch of only the most important features needed for this algorithm. Following, first, to each cell c' of S' we assign a positive half space $H^+(c')$ such that the cell's boundary contains c' and the normal $N(c')$ points towards the interior of $H^+(c')$, see Figure 6.5(a). Second, a cell c of the surface S is classified as follows: For all vertices v of c we search for the closest point in S' and for a cell c' which contains this point, in order to then check whether v is an element of the half space $H^+(c')$. If at least two of three vertices of the cell c are elements of positive half spaces we classify c to be a cell on the *upper* side of the surface S . Since we are dealing with tetrahedra (Tet4 elements), and hence have an odd number of vertices for each surface triangle, the thus obtained classification is unambiguous.

Figure 6.5(b) illustrates the reason why we need to iterate over at least two of three vertices of a cell: In the ideal world case, the segmented MV geometry S' would – after application of the above three general purpose MSML operators – perfectly be nested inside the volume mesh V . The segmentation S' would hence be completely covered by the surface S of the volume mesh V , as it is shown in Figure 6.5(a). In this case, we would not need to iterate over at least two of three vertices of a cell, but instead only check one vertex of each cell, since the position of a cell's vertex would directly determine the position of the cell. However, locally, the surface

S can have irregularities and sharp edges, for instance, in case the MSML-based volume mesh generation operator is too coarse. A possible result is schematically depicted in Figure 6.5(b). Therefore, a cell on the upper side of the leaflet can have vertices that are not elements of positive half spaces. Hence, in order to guarantee for a reliable and robust classification, we need to iterate over at least two of three vertices for each cell.

Figure 6.6(a) shows a color-coded representation of the upper and lower cells of an MV surface mesh that was generated with the proposed method. By assigning two different IDs to respectively the upper or lower surface cells, we obtain the corresponding surfacewise Neumann BC data structure (as listed in Section 5.2.1, paragraph “The Simulation Scenario Input and Control Interface”), which allows for the blood pressure to accordingly act on the MV during systole or diastole.

6.2.1.2 Facetwise Penalty Contact BCs (Leaflet Contact): Identification of Anterior and Posterior Leaflets

In order to determine whether a cell on the surface mesh S belongs to the anterior or posterior leaflet of the MV, we make use of the vertex IDs given by the segmented MV geometry S' as illustrated in Figure 6.4.

Again, by c we denote a cell of the surface mesh S . Then, for all vertices v of c we search for the closest point p' in S' . As described above and illustrated in Figure 6.4, the ID of this point p' assigns it to either the anterior or the posterior leaflet or to the annulus. Thus, if p' is assigned to one of the leaflets, we label the vertex v accordingly, too. If p' is an element of the annulus, we search for the next closest point to v in S' which is not p' , and possibly repeat this process until we find a closest point to the vertex v that is an element of one of the leaflets.

Figure 6.6(b) shows a respectively color-coded representation of anterior and posterior cells on the upper side of an MV surface mesh, as obtained by means of the proposed method. Again, by assigning two different IDs to respectively the anterior or posterior leaflet surface cells, we obtain the required information for the surfacewise contact BC data structure (as listed in Section 5.2.1, paragraph “The Simulation Scenario Input and Control Interface”), which allows for the efficient handling of contact between the two MV leaflets during the MV closing process.

6.2.1.3 Pointwise Neumann BCs (Chordae tendinae): Computation of the Chordae Attachment Points and Chordae Force Directions

As motivated in Section 4.2.1.1, we account for the pulling action of the chordae tendinae on the leaflets by imposing pointwise Neumann BCs. In our model, we distinguish between the so-called *primary chordae*, which are attached to the lower rim of the leaflets, and the *secondary chordae*, which branch off the primary chordae and are attached to the surface area of the lower leaflet sides.

Therefore, first, we present a method to distribute the attachment points of the primary chordae on the lower rim of the leaflets, and let the chordae-induced pulling forces point from these attachment points towards representations of the tips of the papillary muscles, see again Figure 4.6. Second, we will suggest an algorithm to distribute a set of secondary chordae attachment points on the lower leaflet sides, again along with a representation of their

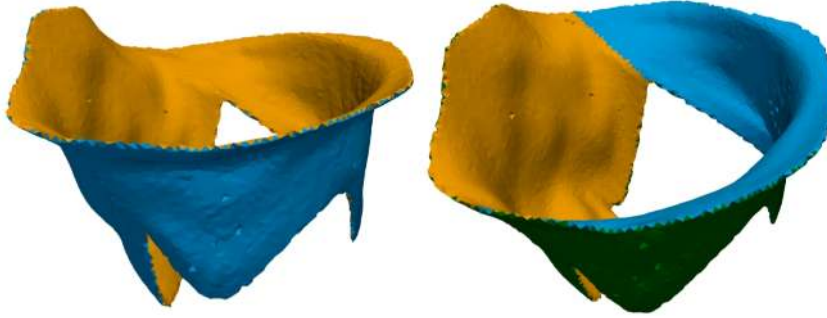


Figure 6.6: Color-coded representation of the cells on the MV leaflet surfaces. **(a)** Identification of upper and lower leaflet sides. **(b)** Identification of the anterior and posterior leaflets. Both meshes were generated through application of our dedicated MVR simulation preprocessing operators.

respective pulling force vectors, which point towards the papillary muscles and to the primary chordae.

We start with the *primary chordae*: Therefore, in order to equally distribute the chordae attachment points along the lower rim of the MV leaflets, we first have to filter out a subset of cells of S which constitute the lower leaflet rims. Using the results from the above described algorithm, which enables the distinction of cells on the upper and lower leaflet sides (see again Section 6.2.1.1), we let C' be the set of cells on the lower side of the MV surface that are neighbor to a cell on the upper leaflet side. The set C' is depicted in Figure 6.7(a). Following, we search for a cell in the set C' such that one of its vertices has a minimal z -coordinate among all the vertices of cells in C' (with z -orientation as shown and defined in Figure 6.4). The lower rim C of the MV leaflets surface then is given as the *connected component* of cells in C' that contains the previously specified cell with minimal z -coordinate. In order to compute this connected component, we first initialize C as the set that contains this previously specified cell with the minimal z -coordinate, and then recursively add cells to C which are in C' and neighbor a cell in C . See Figure 6.7(b) for an illustration of the thus obtained lower rim.

Having computed the lower rim C of the MV leaflet surface, we now want to equally distribute – as suggested by Mansi et al. [MVG⁺12] – the primary chordae’s attachment points on the set C , i.e., along the vertices on the lower rim. In order to do so, we use the so-called *Farthest-First Traversal* algorithm [Gon85], which is a greedy algorithm that computes a *2-approximation* to the optimal solution of the so-called k -center clustering problem. k -center clustering problems are known from combinatorial optimization in graph theory, and were originally used for finding k clusters that minimize the maximum diameter of a cluster, see, e.g., the work by Gonzalez [Gon85]. The problem reads as follows:

Given a metric space X , a finite subset $S \subseteq X$ and a positive integer k , find a subset $T \subseteq S$, the center set, with $\#T = |T| = k$, which is a solution to the problem

$$\min_{T \subseteq S, |T|=k} \max_{p \in S} d(p, T),$$

where again $d(\cdot, \cdot)$ denotes the Eukclidean distance.

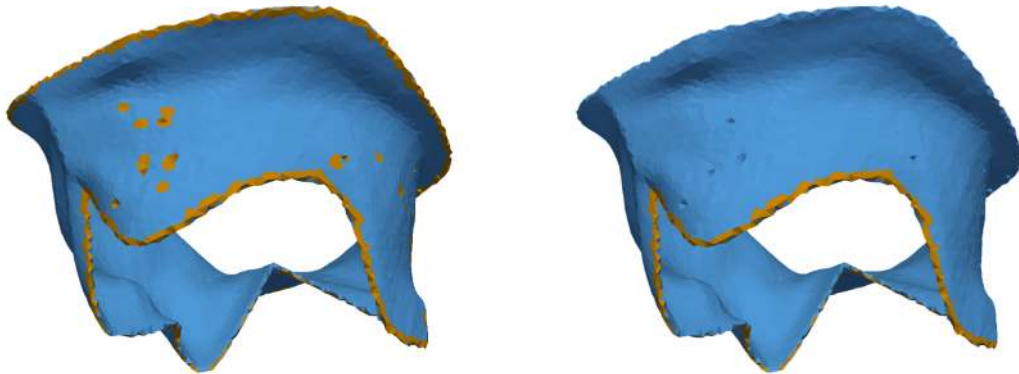


Figure 6.7: Illustration of the stepwise results from the described algorithm: **(a)** Color-coded representation of cells on the MV’s lower leaflet sides that neighbor a cell of the MV’s upper leaflet sides. One obtains both the upper and lower rim as well as some irregularities due to coarse volume meshing. **(b)** Finally, the lower rim of the leaflets is obtained via the connected component of the cell with minimal z -component.

We use a greedy algorithm, since there exist no methods that solve this optimization problem while still running in polynomial time. Hence, applying the Farthest-First Traversal algorithm to the lower rim of the MV surface, we obtain a uniform distribution of the (primary) chordae attachment points. See Figure 6.8 for a thus-obtained distribution of 20 points on the lower rim of the MV surface.

After knowing the set of chordae attachment points, what is missing, is the representation of the chordae pulling forces towards the papillary muscles. However, since currently employed ultrasound imaging techniques do not allow for the segmentation of the actual papillary muscle tips, in our current implementation, the chordae pull forces are directed towards two points below the two leaflets which are to represent these papillary muscle tips in the left ventricle, see Figure 4.6. The *positions* of these two points are heuristically motivated and correspond to the average tip locations in terms of distance from the leaflets and center of gravity with respect to the curvature of the leaflets’ lower rims. These seem to be reasonable assumptions from the surgical point of view. Ongoing, once enhanced imaging techniques allow for a time-resolved identification of the entire left ventricle and for a time-resolved patient-individual localization of the papillary muscle tips, this information can, of course, be incorporated into our model and into the respective pointwise Neumann BC data structure, too, such that the current preliminary representation can be replaced. Apart from the positions of the papillary muscle tips, the behavior of the chordae tendinae is also strongly influenced through their elastic nature. Hence, in order to reproduce the *elastic* behavior of the chordae once they are stretched and fully elongated [WS12], and to hence reproduce their damping effects, we scale the pulling forces by means of a logistic growth function. This logistic growth function grows with the distance between the papillary muscle tips and the leaflets’ lower rims, being zero as long as the chordae are not fully stretched, cf. Section 4.2.1.1. Figure 6.9 shows an illustration of the pointwise Neumann BC data structure, as obtained with the described algorithm.

Please note, that further details on the implementation and on the underlying algorithms

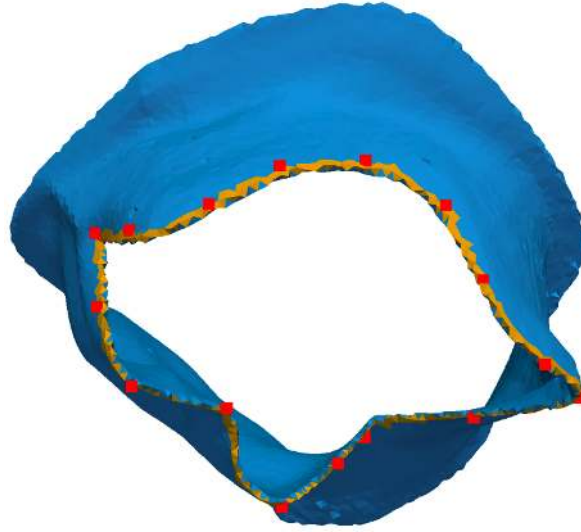


Figure 6.8: Illustration of 20 equally distributed chordae attachment points along the MV leaflets' lower rim. The distribution was computed by means of the Farthest-First Traversal algorithm as described in the text.

can be found in our works [SKS⁺15] and [SKS⁺16].

Next, we look at the representation of the *secondary chordae*: These are attached on the one end to the lower leaflet sides' surface, and on the other end point to the primary chordae or to the papillary muscles. The upper ends of the secondary chordae are located on the lower sides of the leaflets, however, based on observation, only up to a certain distance from the natural annulus. Thus, in order to filter out this *band* around the natural annulus, i.e., around the upper rim of the MV leaflets, where there are no secondary chordae attachment points, we proceed similarly to what was explained above. Starting from the situation depicted in Figure 6.7(a), we now search for a cell in the set C' such that one of its vertices has a maximal z -coordinate among all the vertices of cells in C' , and then find the upper rim \tilde{C} of the MV leaflets (i.e., a representation of the natural annulus) as the *connected component* of cells in C' that contains the previously specified cell with maximal z -coordinate.

In order to obtain a *band* of cells around the upper rim \tilde{C} , we repeatedly add the neighboring cells of all cells in \tilde{C} to the set \tilde{C} , which thus continuously broadens around the natural annulus. Repeating this step seven times empirically showed to be suitable for our purposes, thereby assuming that the average cell size as obtained from the MSML-based volume mesh generator stays approximately the same. We assign a new ID to the cells in the set \tilde{C} , i.e., to the cells that are part of this band. This ID differs from the IDs assigned to upper/lower or anterior/posterior leaflet sides. See Figure 6.10 for an illustration of the described band, which is color-coded in orange.

Based on the thus obtained remaining parts of the lower leaflet sides (depicted in orange in Figure 6.10), we are now left with the task of distributing a certain number of secondary chordae attachment points across these leaflet parts. Hence, again applying the Farthest First Traversal algorithm – now, on the set of remaining vertices – we obtain a set of uniformly

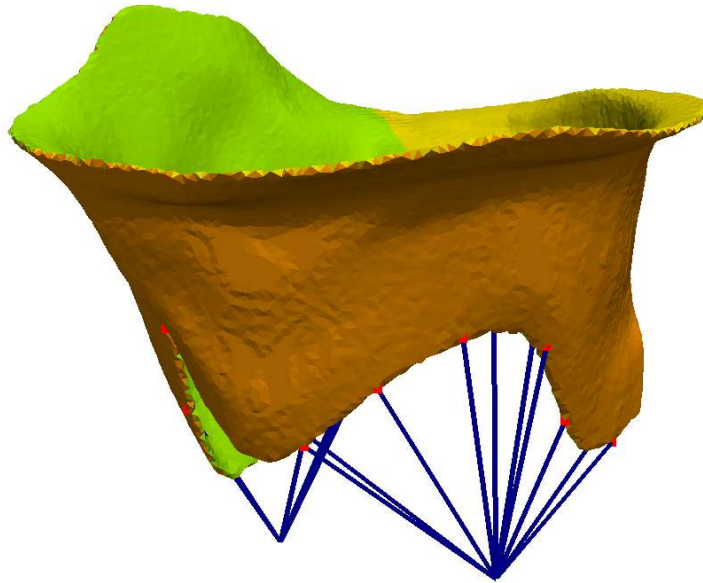


Figure 6.9: Illustration of the primary chordae tendinae, i.e., a visualization of our pointwise Neumann BC data structure, as obtained with the described operator.

distributed secondary chordae attachment points, see Figure 6.10.

Lastly, as above, we compute the vectors from these attachment points to the two points that approximate the tips of the papillary muscles in order to obtain analogous representations of the pulling force directions of the secondary chordae. An illustration of the complete setup of our pointwise Neumann BCs, especially including the secondary chordae, can be seen in Figure 6.10.

The overall simulation-compatible result of this operator is hence finally obtained by transferring the respective (primary and secondary) chordae attachment points and their corresponding pulling force vectors (along the strings) into the pointwise Neumann BC data structure as presented in Section 5.2.1, paragraph “The Simulation Scenario Input and Control Interface”.

Please note, that the computation of a band around the natural annulus, and in particular its classification with a separate ID (which is different from the IDs that classify upper or lower, and anterior or posterior leaflet sides), is also needed in order to ensure that Dirichlet BCs (along the MV annulus) and Neumann BCs (across the MV leaflets) cannot be active on the same element: As explained in Section 4.2.1.1, at boundary interface regions, where Dirichlet boundaries get to meet effective Neumann boundaries, the solution of the contact elasticity problem may exhibit a certain singular behavior due to the potentially unbounded gradient or stress tensor, respectively. We therefore suggested the introduction of a layer of zero Neumann boundary Γ_N between the Dirichlet boundary region Γ_D and the effective Neumann boundary region $\Gamma_N^{\neq 0}$ such that this potential problem can be prevented, see again Figure 4.8.

Another positive side effect of the described band is that it is also excluded from the anterior and posterior leaflet cells over which a contact search (see Section 5.2.2) has to be performed. Given the size of the band, we thus achieve another 12–13% speedup with respect to the computation time requirements.

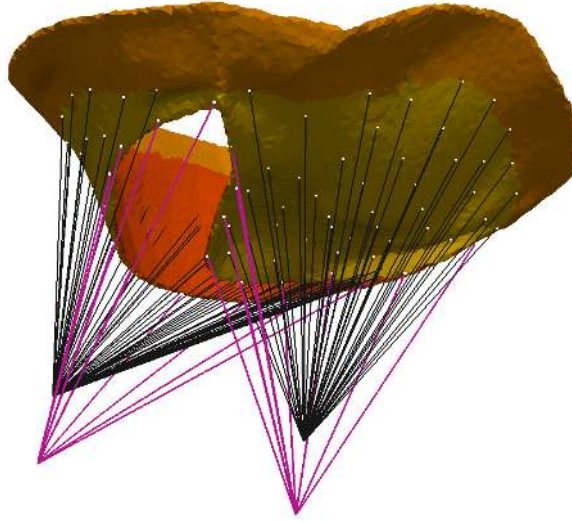


Figure 6.10: Illustration of the complete set of pointwise Neumann BCs, i.e., of primary (thick violet) and secondary (thin black) chordae tendinae, as obtained with the described algorithms. Also, the band around the natural annulus is color-coded in orange, and the upper/lower leaflet sides are painted in red/yellow, respectively.

6.2.1.4 Pointwise Dirichlet BCs (Annuloplasty Ring Implantation): Computation of the Displacement Vectors

During the annuloplasty ring prosthesis implantation, selected points x_{natural} on the natural annulus are displaced by means of a suture to selected points x_{imposed} on the artificial annuloplasty ring prosthesis. According to Section 4.2.1.2, in order to represent this process in the biomechanical model, we compute the respective displacement vectors pointing from the natural annulus to the artificial ring prosthesis, and make use of corresponding Dirichlet BCs, see again Figure 2.7.

However, as shown in Figure 6.11(a), through tightening the natural MV tissue along the artificial annuloplasty ring, it commonly occurs that tissue folds are formed between the suture points that are proposed by the prosthesis model in [ELAM⁺15]. If a fold is too large, the leaflet stiffens and the tissue flexibility is decreased, such that it is then recommended to sew additional points to the ring in order to correct the unintended fold formation, see Figure 6.11(b).

Hence, in order to model this procedure using Dirichlet BCs, first, for all prescribed suture points on the segmented MV geometry S' , as in Figure 6.4, we search for a closest point x_{natural}^i on the respective surface mesh S . As explained above, for each suture point on the natural annulus there is a corresponding suture point on the artificial annuloplasty ring prosthesis which we denote by x_{imposed}^i . Obviously, the resulting displacement vectors are then given by $du^i = x_{\text{imposed}}^i - x_{\text{natural}}^i$. See Figure 6.12(a) for an illustration.

Using our MVR simulation preprocessing operator, we also want to prevent the formation of unintended tissue folds, and therefore proceed as follows: First, we compute the average point-to-point distance of all pairs of consecutive suture points on the natural annulus. Second, we check if one or more of the point-to-point distances of such pairs differs from the overall

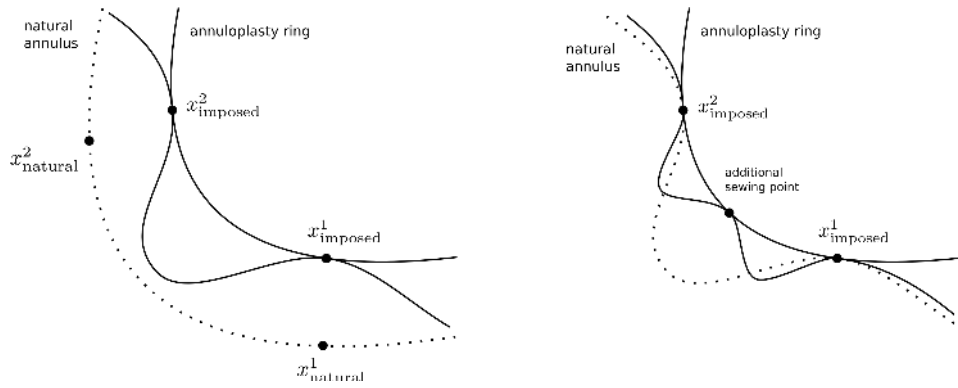


Figure 6.11: Illustration of the annuloplasty ring implantation during MVR, showing the natural annulus and the artificial annuloplasty ring with a set of suture points. The dotted line indicates the position and shape respectively of the natural annulus before MVR surgery or before the application of the fold correction. **(a)** Through tightening the tissue to the artificial ring prosthesis, a fold forms between two suture points. **(b)** Via an additional suture point, this fold formation can be corrected.

average distance by more than 10%, which empirically showed to be reasonable threshold value. If so, we add additional suture points to natural annulus and artificial annuloplasty ring, see Figure 6.11**(b)**. Therefore, as shown in Figure 6.12**(a)**, we compute the midpoints of the respective line segments that connect the suggested suture points on natural annulus and artificial ring, and denote these by $x_{\text{natural}}^{\text{virtual}}$ and $x_{\text{imposed}}^{\text{virtual}}$. As shown in Figure 6.12**(b)**, the added auxiliary suture point x_{aux} then is a closest point to $x_{\text{natural}}^{\text{virtual}}$ on the surface of the actual MV mesh S with the corresponding displacement vector $du_{\text{aux}} = x_{\text{imposed}}^{\text{virtual}} - x_{\text{natural}}^{\text{virtual}}$.

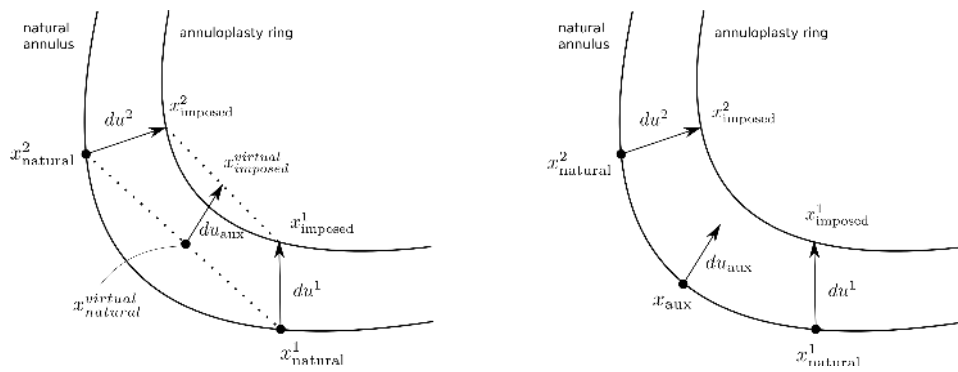


Figure 6.12: Draft of the displacement vector computation with respect to the natural MV annulus and the artificial annuloplasty ring prosthesis and the respective suture points. **(a)** Initial computation of additional auxiliary Dirichlet BCs. **(b)** Representation of additional Dirichlet BCs after subsequent closest point search.

Finally, the computed suture points and their respective displacement vectors are transferred

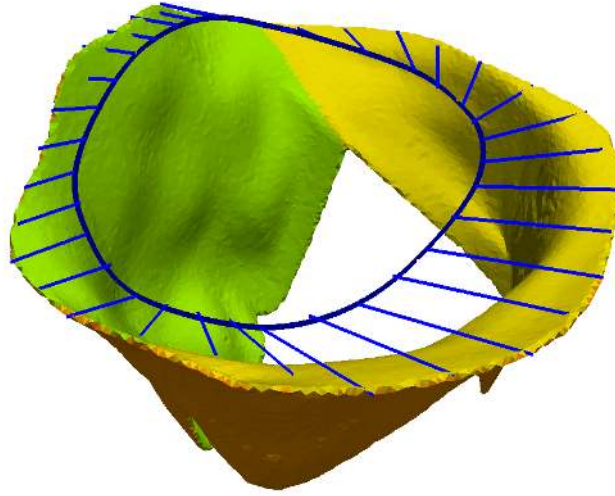


Figure 6.13: Illustration of the pointwise Dirichlet constraints given through the implantation of the annuloplasty ring prosthesis. The visualization was obtained from the data structures generated with our dedicated pointwise Dirichlet BCs simulation preprocessing operator as described in the text.

into the pointwise Dirichlet BC data structure (as listed in Section 5.2.1, paragraph “The Simulation Scenario Input and Control Interface”), in order for the simulation to facilitate the MVR annuloplasty ring implantation process. Figure 6.13 shows a visualization of the obtained pointwise Dirichlet BC data structure.

6.2.2 Integration of the MVR-specific Operators into the MSML

Comprehensive, Patient-specific MVR Simulation Scenario Setup. Combining the set of MVR simulation preprocessing operators presented just above in Section 6.2.1, we obtain an MVR simulation preprocessing pipeline for setting up comprehensive, knowledge-based, patient-specific MVR surgery simulation scenarios. The execution of this pipeline facilitates the holistic information processing of both medical patient data and of surgical MVR expert knowledge.

It is to be emphasized that the presented set of operators is not only motivated by and implemented with respect to the MV anatomy and the surgical procedure of an MVR. It also incorporates, analyzes and processes all available patient-specific information, such as 4D-TEE-US imaging-based segmentations of the MV (as obtained from Engelhardt et al. [ELAM⁺15]), material parameters (as listed for classified average patient populations in the work of Mansi et al. [MVG⁺12]), and the surgical strategy (with respect to the implantation of a suitable ring prosthesis, as obtained from Schoch et al. [SPW⁺16]). See Figure 6.14 for a visualization of two exemplary MVR simulation scenarios that were obtained using the described series of MVR simulation preprocessing operators. Among others, Figure 6.14 illustrates the MV mesh along with the different cell IDs, the representations of the chordae tendinae and the annuloplasty ring implantation constraints, thus covering the entire set of boundary conditions.

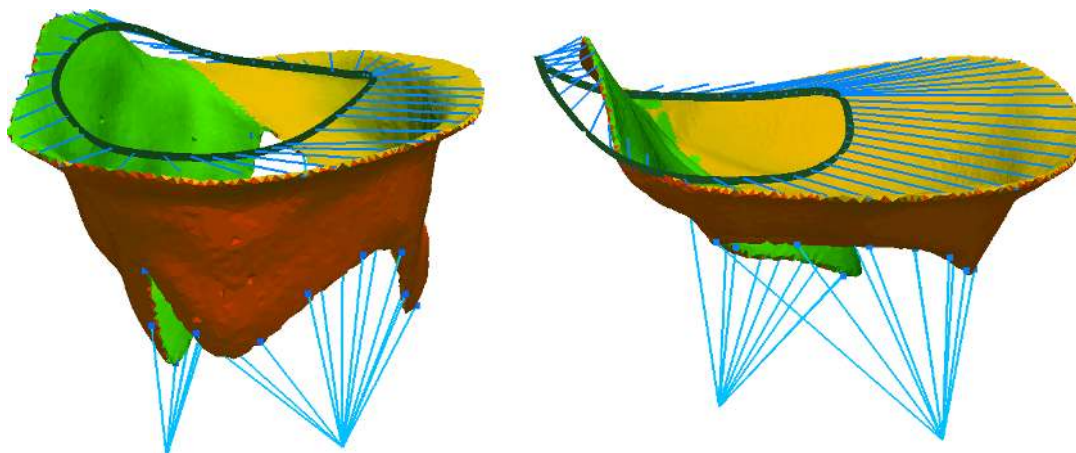


Figure 6.14: Visualization of two exemplary MVR simulation scenario setups with the respective representation of the BC data structures derived by means of the above presented series of MVR simulation preprocessing algorithms.

Having executed the set of MVR simulation preprocessing operators consecutively as part of a comprehensive operator chain, and thus having produced the corresponding simulation input data, we have fully defined and set up a comprehensive, patient-individual MVR simulation scenario along with the required input (as described in Listing 5.3 in Section 5.2.1). We can thus directly run our HiFlow³-based MVR simulation application (see Section 5.2), which in turn allows for the surgeon to investigate – *before* the surgical intervention – on the virtually simulated behavior of the MV. We emphasize that with the visualization of both the simulated post-MVR behavior of the MV and of the corresponding underlying biomechanical MVR model as in Figure 6.14, transparency and hence the surgeons’ understanding is enormously fostered: the operating clinician may easier evaluate *how* the post-surgical MV behaves, and *why* (i.e., with respect to which natural or MVR-induced constraints) it does so.

Full Automation of the MVR Simulation Preprocessing through the MSML. In Section 6.1, we presented the *Medical Simulation Markup Language* (MSML). It has been motivated how the MSML modeling approach aims at setting up patient-specific, FEM-based biomechanical models from tomographic data, and how it simultaneously allows to define the model and simulation features *and* the complete biomechanical modeling and simulation workflow, thereby also acting as a middleware between all tools used.

Building up on this work, we also integrated the above set of dedicated MVR simulation preprocessing operators into the framework of the MSML, according to the procedure described in Section 6.1. This way, all operators-deduced data are transferred into MSML-internal data structures (such as geometry information, mesh properties, material IDs, boundary condition specifications, etc., as presented in Section 6.1), and hence available for further usage and processing, e.g., by means of an MSML simulation engine exporter.

Going more into the details, our overall pipeline of MVR simulation preprocessing operators incorporates patient-specific MV geometries that are processed into FEM-suitable meshes. It takes account for blood pressure profiles via dynamic Neumann BCs, as well as for upcoming

contact between the two leaflets during the cardiac cycle via Contact BCs, and finally, it also considers the virtual implantation of the annuloplasty ring prosthesis via Dirichlet BCs.

Being retrieved by the corresponding operators and processed via MSML data structures, these pieces of information are then handed over to the HiFlow³ simulation exporter. However, in order for this exporter to produce HiFlow³-compatible `hf3.xml` MVR simulation scenario files, that can be run by our dedicated, purpose-optimized HiFlow³-based MVR simulation application (as presented in Section 5.2.1, paragraph “The Simulation Scenario Input and Control Interface”), we had to adapt and extend the general *standard HiFlow³ exporter* such that it is capable of processing and producing MVR-specific data structures. To be precise, as opposed to the above described standard base HiFlow³ exporter, which was designed for the setup of general (non-specific) biomechanical FEM simulations, we have implemented the *HiFlow³ Mitral exporter*, which is a derived class and which inherits from the standard HiFlow³ exporter. Being designed for the specific purpose of MVR simulations only, the HiFlow³ Mitral exporter produces `hf3.xml` MVR simulation scenario files (as in Listing 5.3), that can directly be run by our purpose-optimized HiFlow³-based MVR simulation application (as presented in Section 5.2.1). Along with the `hf3.xml` MVR simulation scenario input files, it also produces and links two other files, a `mesh.inp` file and a `bcdata.xml` file, in order to describe input mesh and boundary conditions.

Listing 6.4 shows a typical `msml.xml` file, i.e., an MSML-based description of a biomechanical MVR modeling workflow for an MVR simulation scenario that can be executed by means of our HiFlow³ MVR simulation application. Among others, it starts with the definition of the *I/O variables* (lines 5 ff), goes on with a chain of *workflow operators* (lines 13 ff), which also details how the respectively produced data is further-processed by some next operators, and finally describes the *MSML scene* (lines 39 ff) and the *MSML environment* (lines 82 ff), i.e., how materials, boundary conditions, and the overall simulation scenario are composed. A such `msml.xml` file can be executed and interpreted by the MSML using the command

```
python src/msml.py exec mvrScenePipe.msml.xml -e mitral -o outfolder/
```

where the `-e mitral` part addresses the specific HiFlow³ Mitral exporter.

Listing 6.4: Sample `msml.xml` file describing an MVR simulation and its setup.

```

1 <msml:msml xmlns:msml="http://sfb125.de/msml"
2     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://sfb125.de/msml">
4
5 <variables>
6   <var name="input_mv_surfmesh"
7       value="MVR_PatientXY_Segmentation.vtp"
8       logical="TriangularSurface" physical="vtp" role="input" />
9   <var name="input_mv_ring"
10      value="MVR_PatientXY_AnnuloplastyRing_PhysioII_Size32.vtp"
11      logical="object" physical="vtp" role="input" />
12 </variables>
13 <workflow>
14   <SurfaceToVoxelDataOperator id="aSurfaceToVoxelsOperator"
15       surfaceMesh="\${input_mv_surfmesh}"
16       targetImageFilename="mvImage.vti"

```



```

17         accuracy_level="10" smoothing="1" />
18 <vtkMarchingCube id="aVoxelToContourOperator"
19     image="\${aSurfaceToVoxelsOperator}"
20     outFilename="mvVolumeSurface.vtk" isoValue="90" />
21 <CGALMeshVolumeFromSurface id="aVolumeMesher"
22     meshFilename="mvVolumeMesh3D_CGAL.vtk"
23     surfaceMesh="\${aVoxelToContourOperator}"
24     preserveFeatures="false" facetAngle="20" facetSize="0.4"
25     facetDistance="0.3" cellRadiusEdgeRatio="3" cellSize="0.4"
26     odt="false" lloyd="false" pertube="false" exude="false" />
27 <vtuToHf3inpIncMVmatIDsProducer id="aVtuToHf3inpIncMVmatIDsProducer"
28     volume="\${aVolumeMesher}" surface="\${input_mv_surfmesh}"/>
29 <mvGeometryAnalytics id="anMvGeometryAnalyzer"
30     surface="\${aVolumeMesher}" ring="\${input_mv_ring}" />
31 <mvrBCdataProducer id="aBCdataForMVRsimProducer"
32     volumeMesh="\${aVolumeMesher}"
33     surfaceMesh="\${input_mv_surfmesh}"
34     ring="\${input_mv_ring}" />
35 <mvrBCdataExtender id="aBCdataForMVRsimExtender"
36     volumeMesh="\${aVolumeMesher}"
37     surfaceMesh="\${input_mv_surfmesh}" />
38 </workflow>
39 <scene>
40     <object id="mitralvalve">
41         <mesh>
42             <linearTet id="mvVolMesh"
43                 mesh="\${aVtuToHf3inpIncMVmatIDsProducer.inpMeshFile}" />
44         </mesh>
45         <material>
46             <region id="mvMaterial"
47                 indices="\${aVtuToHf3inpIncMVmatIDsProducer.tet_material}">
48                 <materialID matID="10" />
49                 <linearElasticMaterial youngModulus="2350" poissonRatio="0.488" />
50                 <mass name="stdMass" massDensity="1.000" />
51             </region>
52         </material>
53         <constraints>
54             <constraint name="bodyConstraint" forStep="\${ring_implantation}">
55                 <displacedPointsConstraint
56                     points="\${aBCdataForMVRsimProducer.points}"
57                     displacements="\${aBCdataForMVRsimProducer.displacements}" />
58             </constraint>
59             <constraint name="bodyConstraint" forStep="\${swing_off_pause}">
60                 <displacedPointsConstraint
61                     points="\${aBCdataForMVRsimProducer.points}"
62                     displacements="0.0,0.0,0.0" />
63             </constraint>
64             <constraint name="bodyConstraint" forStep="\${cardiac_cycle}">
65                 <displacedPointsConstraint
66                     points="\${aBCdataForMVRsimProducer.points}"
67                     displacements="0.0,0.0,0.0" />
68                 <mvGeometryConstraint
69                     mvGeometry="\${anMvGeometryAnalyzer}" />
70                 <surfacePressureConstraint
71                     indices="17,18,20" pressure="timedependent_wiggersprofile" />

```

```

72     <pointwiseForceConstraint
73         points="\${aBCdataForMVRsimExtender.points}"
74         forces="\${aBCdataForMVRsimExtender.forces}" />
75     </constraint>
76 </constraints>
77 <output>
78     <displacement id="mitralvalve_disp" timestep="1" />
79 </output>
80 </object>
81 </scene>
82 <environment>
83     <solver linearSolver="CG" preconditioner="SGAUSS_SEIDEL"
84         hf3_chanceOfContactBoolean="1"
85         processingUnit="CPU" numParallelProcessesOnCPU="0"
86         timeIntegration="Newmark"
87         RayleighDampingMass="0.5" RayleighDampingStiffness="0.5" />
88 <simulation>
89     <step name="ring_implantation" dt="0.05" iterations="100"
90         visualizeSimResultsEvery1inXtimesteps="10" />
91     <step name="swing_off_pause" dt="0.05" iterations="100"
92         visualizeSimResultsEvery1inXtimesteps="10" />
93     <step name="cardiac_cycle" dt="0.001" iterations="10000"
94         visualizeSimResultsEvery1inXtimesteps="100" />
95 </simulation>
96 </environment>
97
98 </msml:msml>

```

It has to be emphasized, that – via the presented setup of our MSML-based MVR simulation preprocessing pipeline – we allow for the intelligent, patient-individual composition of `hf3.xml` MVR simulation scenario input files, as presented in Listing 5.3. The resulting simulation scenario input files fully describe the entire simulation scenario, and hence act as our simulation scenario control interface; see again Section 5.2.1. Moreover, in addition to *patient-specificity* and to the *holistic information processing*, by using the MSML as a *middleware* between all tools and operators in the MVR simulation preprocessing workflow, we allow for the *full automation* of the MVR simulation scenario setup process.

As opposed to simply implementing a combination or pipeline of single scripts and operators, we allow via the MSML for an easily accessible MVR simulation control interface, with respect to which one does not need to care about input/output, transitional and interim data, file formats, shared operator or pipeline activities, and compatibility of data structures anymore. Compared to previous systems or to systems without any simulator support, this is an important *usability improvement*. We assume that, once installed in the OR, our system has the potential to substantially support clinicians due to the easy and automated usage of simulations in surgery assistance environments. Also, with our development flexibility is maintained throughout the complete workflow, and complex tools that have once been integrated can be accessed and used via a general interface, both for clinicians and for developers. For more information, we refer to our works [SKS⁺15] and [SKS⁺16].

Summarizing, through the work presented in this section, we support minimally-invasive MVR surgery by providing MVR simulation scenario setups for our HiFlow³-based FEM soft tissue simulation application (see Chapter 5), which in turn simulates the behavior of the patient-individual MV subject to natural forces during the cardiac cycle after an MVR.

Going beyond other previously presented works (see again the state-of-the-art in the context of surgery simulation preprocessing as presented in Section 3.1.5.2), with our work [SKS⁺16] we allow – for the first time – for the fully automated, patient-specific, comprehensive simulation scenario setup through a pipeline of dedicated information processing and data analytics operators that are integrated into the framework of the MSML.

The comprised operators intelligently make use of the available surgical MVR knowledge and allow for the analysis, interpretation and processing of patient-specifically given medical data (consisting of both images and parameter data). The resulting simulation scenarios are *comprehensive* (or at least arbitrarily extensible) with respect to a mature underlying biomechanical MVR model (as described in Section 4.2) and also *patient-specific* with respect to the patients' image and parameter data.

We refer back to Figure 6.3 for an illustration of the concept of our MVR simulation preprocessing workflow. It is to be pointed out that the input data (*left block*) for the MSML-based MVR simulation preprocessing pipeline is managed by means of a *cognitive applications pipeline* and via the *cognitive system and data architecture*, which will be presented in Chapter 7.

Closing, we remark that in current MVR surgery, surgeons largely depend on their own experience and on an in-depth surgical expert knowledge. Extending this, our MSML-based MVR simulation preprocessing workflow represents a *usability improvement* which simplifies the usage of surgery simulations in the OR and certainly fosters the acceptance of simulation-enhanced surgery assistance through allowing for providing surgeons with adequate, patient-individual, comprehensive MVR simulation scenarios fully automatically.

In the following Section 6.3, we will deal with the question of how the simulation results are *post-processed* in order for surgeons in the OR to maximally benefit from them, and in order for the results to be really helpful and easy to understand.

6.3 Simulation Postprocessing for MVR Surgery Assistance

The principal benefits for the operating heart surgeon from an MVR soft tissue simulation which simulates the annuloplasty ring implantation process and a post-operative cardiac cycle are basically threefold:

The most obvious benefit resulting from a biomechanical soft tissue simulation of the MV is the opportunity to virtually assess the overall *deformation behavior* of the MV after MVR when the virtually manipulated valve is again subjected to natural forces. Hence, *visualization* of the simulation results is imperative for a good understanding of the 3D morphology. This being said, during the post-operative cardiac cycle, the most important measurable anatomical characteristic of interest is the competence of the MV closing behavior, which is usually measured by means of the *size of the coaptation zone*; see again Figure 2.3.

Going beyond visible tissue deformation properties, surgeons are also interested in the *post-surgical stress behavior* during the valve opening and closing process. The natural stress dis-

tribution is commonly altered after the surgical manipulation due to the annuloplasty ring implantation. Hence, to provide surgeons with additional information on critical stress values, a stress analysis (based on the von Mises stress measure) is often included in the postprocessing visualization step.

In this section, which describes our *MVR simulation postprocessing component*, we first outline the analysis of the coaptation zone, go over to the evaluation of the von Mises stress distribution, and conclude with a computationally lean visualization of the deformation behavior of the MV throughout the cardiac cycle which includes a color-coded projection of the von Mises stress values on the MV leaflet surfaces.

Coaptation zone computation: In addition to a purely visual assessment of the motion and deformation behavior of the MV, the size and connectivity of the coaptation zone as described in Section 2.1 and depicted in Figure 2.3 allows to even better quantify the competence of the MV closing functionality.

In order to compute the size of the coaptation zone for a simulated post-surgical MV with a specific ring implantation, we stick to our notion of *leaflet contact* as defined in Section 4.2.1: contact is established as from a gap distance d_{gap} between the two leaflets of $\varepsilon_{\text{tol}} = 1.5$ mm and lower (with ε_{tol} according to (eq. 4.98)).

Almost trivially, we hence need to sum up the areas of all element facets on the leaflet surfaces that have established contact with respect to ε_{tol} , and provide the surgeon with this information for respectively every time step during the simulated post-surgical cardiac cycle. For easier comparison of the leaflet coaptation competence for different potentially suitable annuloplasty ring prostheses, the surgeon is provided with the maximum value of the coaptation zone across the entire cardiac cycle, which usually represents a fully closed state.

Von Mises stress visualization: In order to analyze the stress behavior of soft tissues in the surgical context, the so-called *von Mises stress* has established as the measure of choice, and today is commonly used as an indicator for material failure [MPW⁺16]. The von Mises stress σ_{vM} is derived from the symmetric Cauchy stress tensor $\boldsymbol{\sigma}$ which was defined in (eq. 4.10). It computes as

$$\sigma_{\text{vM}} = \sqrt{\frac{3}{2}\sigma_{ij}\sigma_{ij} - \frac{1}{2}(\sigma_{kk})^2} \quad (6.1)$$

$$= \sqrt{\sigma_{xx}^2 + \sigma_{yy}^2 + \sigma_{zz}^2 - \sigma_{xx}\sigma_{yy} - \sigma_{xx}\sigma_{zz} - \sigma_{yy}\sigma_{zz} + 3(\sigma_{xy}^2 + \sigma_{xz}^2 + \sigma_{yz}^2)}. \quad (6.2)$$

The scalar von Mises stress value is used to predict the yielding of a material under loading conditions, and the respective *von Mises yield criterion* suggests that the yielding of a material begins when a critical von Mises stress value is reached. For instance, critically high stress values that occur after the implantation of a suboptimal annuloplasty ring prosthesis may indicate the yielding of the tissue, or in worse cases even lead to stenosis, ring dehiscence, or obstruction of the ventricular outflow tract [CAF10, MPW⁺16]. We therefore visualize the von Mises stress on the surface of the MV leaflets and thus give surgeons a virtual *preoperative* hint of possibly negative *postoperative* effects of a specific, ring-dependent surgical manipulation before the actual intervention is performed.

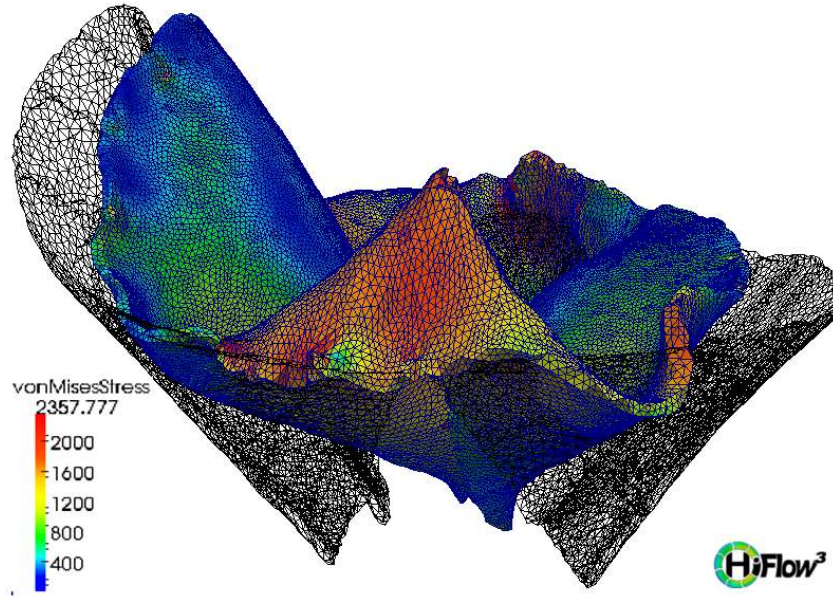


Figure 6.15: Sample MVR simulation scenario result, obtained from our HiFlow³-based MVR simulation application, showing the MV leaflets before (wireframe) and after (colored) the implantation of a specific annuloplasty ring prosthesis. The von Mises stress distribution (in MPa) as computed in our MVR simulation postprocessing component is color-coded and projected onto the MV leaflet surface.

In the following, we depict the von Mises stress computation using VTK methods.

In Section 5.2.1, we have described the structure of the output data resulting from our HiFlow³-based MVR simulation application, which generally consists – for every simulated time step – of a `pvtu` file (as in Listing 5.4) and a series of n_{proc} `vtu` files (as in Listing 5.5).

After application of the HiFlow³-attached `pvtu2vtu` conversion operator, the obtained single `vtu` file (one of which is obtained for every respective simulation time step) contains all required information on the displacement vectors $\mathbf{u} = (u_0, u_1, u_2)$ of the nodes of the MV mesh representation V and on their connectivity. In order to now visualize the deformed configuration of the MV, we use the VTK class `vtkWarpScalar` to successively add the scalar values u_i to the respective arrays of the coordinates of the respective nodes in the mesh V . We thus obtain the vertex coordinates of the deformed MV object.

Thereon-based, the computation of the von Mises stress then entails three steps: First, we use the VTK class `vtkArrayCalculator` to compute an array of displacement vectors for every vertex in V from the scalar arrays u_i . Second, the strain tensor is computed by means of the class `vtkCellDerivatives`, thereby using the previously generated array of displacement vectors, and lastly, the von Mises stress value is computed from the strain tensor according to (eq. 6.2) and to Hooke’s Law (eq. 4.60), again using the class `vtkArrayCalculator`.

Figure 6.15 exemplarily shows a visualization of a sample MVR simulation scenario, where the MV is depicted at a half-closed stage after a virtually simulated ring prosthesis implantation, and where the color-coded von Mises stress distribution is projected onto the MV leaflet surface.

Knowing about the size of our set of simulation results and its large memory consumption on the one hand, and given the computational efforts for the von Mises stress computation for a long series of simulated time steps during a cardiac cycle on the other hand, it is reasonable to have the simulation results immediately processed by the von Mises stress evaluation operator once they are obtained and available from the HPC cluster.

Hence, once available they are converted by means of the above-mentioned `pvtu2vtu` conversion operator and stored in a specific simulation scenario results directory. Employing the Python module `glob`, which returns an iterator of files in a directory using shell pattern matching, to this simulation output directory, we can have the von Mises stress evaluation script run over all simulation results one after the other. Using `iglob` even allows to perform this task on a data-driven basis in real-time as soon as new simulation results are available, as the data is not all stored in one buffer or list in memory, but can be read out one at a time.

This allows to efficiently add the array of von Mises stress values to the `.vtu` files in order for subsequent visualization in the OR on some OR workstation, see next paragraph. Further details on the underlying algorithms can also be found in our work [SKS⁺15].

Lean visualization: Lastly, we want to provide the operating surgeon in the OR with a meaningful visual representation of our simulation results and the thereon-based above described postprocessing analysis. That is, we want to make available the simulated 3D deformation behavior of the MV during a post-MVR cardiac cycle in combination with the color-coded projection of the von Mises stress distribution on the surface of the MV leaflets.

However, when aiming for the visualization of these features on some OR workstation, one has to keep in mind the large memory consumption of our simulation results and the thus-induced long information/data transfer requirements between HPC cluster and OR workstation. Another direct disadvantage resulting from the large data sets is that one is often confronted with the issue of jerks during the visual animation of the 3D simulation results, which are due to the fact that only few 2D images can be rendered per second when the 3D objects are too big.

Hence, a computationally lean visualization is imperative for in-OR-visualization, and the last step of our MVR simulation postprocessing component is dedicated to this purpose:

Resulting from the `pvtu2vtu` conversion operator and from the subsequent application of our von Mises stress evaluation algorithm, we have obtained `vtu` files (one for each simulated time step), which contain both the arrays of the deformed coordinates of the nodes of the MV volume mesh V and the array of the appendant von Mises stress values. Thereon-based, we first employ the VTK filter `cleanToGrid`, which takes the polygonal grid as input and generates a purged, lean, new polygonal grid as output by merging duplicate points, removing unused points, and transforming degenerate cells into their appropriate forms. Next, the application of the VTK filter `extractSurface` facilitates the extraction of the polygons forming the outer surface of the dataset containing the volume mesh. The `extractSurface` filter produces polygonal data as output, optionally with a set of specified attributes for the surface elements. In our case, we are only interested in the von Mises stress attribute, which is hence appended to the extracted MV surface.

The resulting MV surface mesh representation `vtu` file finally occupies only about 20-25% of the size of the above original `vtu` file, and thus guarantees a smooth visualization and rendering process of our MVR simulation scenario results even on relatively weak OR workstations.

6.4 Discussion, Future Intended Work and Outlook

In the previous three sections, we have presented our approach of the Medical Simulation Markup Language (MSML) for simplifying the general biomechanical modeling and simulation workflow (Section 6.1), and exemplarily demonstrated its application for MVR surgery simulation preprocessing (Section 6.2) and postprocessing (Section 6.3).

By means of the clinical MVR application scenario we have shown how the MSML not only allows to describe the biomechanical model, but also how it acts as a middleware between all tools used in the biomechanical modeling pipeline. Also, due to the large amount of MVR-specific needs, we have illustrated its flexibility, extensibility and robustness, as well as its helpfulness in the context of simulation support and generally for the in-OR-usability of surgery simulations.

However, having presented a set of dedicated MSML pipelines in combination with the specific HiFlow³-based MVR simulation application, we are well-aware of the difficult *transferability* and of potentially upcoming issues with respect to the *generalization* of our software solution towards other specific surgical application fields. For instance, even though we designed both the operators and the data structures such that they can be used and employed for different purposes and for different export target simulation engines, we have currently only implemented these operators so they fit the specific interface of the HiFlow³ Mitral exporter.

As already mentioned in Section 6.1, this is a general issue, since generally not all export targets necessarily support the entire set of features and functionalities described in the overall MSML alphabet. We have motivated that, for this reason, we have furnished every MSML exporter with a compatibility tree that lists its compatible nodes. This allows for an easy compatibility checking scheme which ensures the consistency between the MSML description and the respective FE simulation application.

This being said, as a conclusion, it is obvious that in order to exploit the full capabilities of the MSML, e.g., with regard to an evaluation of our MVR pipeline or of our HiFlow³-based MVR simulation application, the respectively missing functionalities would need to be completed such that every MSML operator can produce data for every MSML exporter and for every MSML-linked simulation engine. We denote this requirement and process by *MSMLalization* of the remaining features and interfaces.

Obviously, once a new feature or functionality has once been implemented and made compatible with the MSML for a specific purpose and for a specific export operator, the abstraction, generalization and transfer of the respective feature to other exporters and simulation engines is – in a research context – only achievable up to a certain level as from which on further completion is a rather routine piece of work. Hence, it seems more reasonable to aim at some sort of ‘specific generalization’ for a limited set of a few sample surgery scenarios only.

With the presented MSML-based MVR simulation pre- and post-processing pipelines, we have demonstrated the usage and benefits of the MSML for biomechanical soft tissue simulations, its flexible and robust model description, and its capability to act as a middleware between different tools in the complex biomechanical modeling workflow. We have explained above that further material models, operators and exporters can be integrated into the MSML simply by adding an `xml` file (as in Listing 6.3) to the MSML alphabet. In the following, we will look at three fields towards which the MSML may be extended in the future.

MSML goes CFD: So far, the MSML only covers purely solid mechanics-based phenomena, however, one may of course think of adding to the MSML a set of models for the simulation of *flow phenomena* that are based on Computational Fluid Dynamics (CFD). An initial concept has already been presented in our work [SSS⁺14]:

In the context of cardiovascular treatment planning, the MSML may be extended to describe material models and inflow/outflow boundary conditions, as well as specific time-dependent vector fields, to thus generate a biomechanical model for, e.g., the hemodynamics in the aorta. Looking at the landscape of current CFD assistance and modeling tools, for the purpose of making the MSML suitable for CFD modeling and simulation, the definition of an interface to the *Vascular Modeling Toolkit* (vmtk) [EMSH97] seems promising. vmtk-based workflow features for the segmentation of vascular segments from medical images or for geometric analysis and I/O tools can thus immediately become accessible and controllable via the MSML.

MSML goes UQ: Similarly, the integration of methods for *Uncertainty Quantification* (UQ) into simulation toolkits will become more and more important. This especially holds for surgery simulations, where simulation input data (such as medical images or material parameters) are generally subjected to noise and uncertainties, and where the reliability of simulation results has to be assessed and quantified for surgical usage. Therefore, it is obviously reasonable to define and implement an interface to the MSML for some (non-intrusive) UQ toolkit, which preferably is capable of being applied to arbitrary simulation pre- and postprocessing components, and which handles the simulation itself as a black box.

For instance, one may assume the two Lamé material parameters λ and μ to be subject to uncertainties. Hence, given a probability distribution for these two parameters, which enter the biomechanical model description from the MSML-controlled UQ module via the MSML HiFlow³ Mitral exporter, which in turn is part of the simulation preprocessing component, one can run a series of MVR simulation scenarios which covers this probability distribution. The simulation results can then again be post-processed by the MSML-controlled UQ module within the simulation postprocessing component, in order to provide information about the most probable simulation scenario result or about a specific range or potential variation of the MV tissue behavior. See Figure 6.16 for an extended illustration of the schematic biomechanical modeling workflow which – in comparison to Figure 6.1 – additionally considers UQ for the setup of the biomechanical model and simulation scenario via the pre- and post-processing components.

There are several established open-source toolkits which provide UQ functionalities. Among others, we consider *ChaosPy* [FL15] an important numerical tool for performing UQ using polynomial chaos expansions and advanced Monte Carlo methods. Besides that, *Dakota* [ABB⁺15] has proven suitable for many engineering purposes, as it provides a flexible, extensible interface and contains algorithms for UQ with sampling, reliability, stochastic expansion, and epistemic methods, as well as for parameter estimation with nonlinear least squares methods, and for sensitivity/variance analysis with design of experiments and parameter study methods.

Interestingly, even though there are plenty of validated UQ toolkits and evaluated methods freely available, UQ does not yet seem to be commonly employed for surgery simulations.

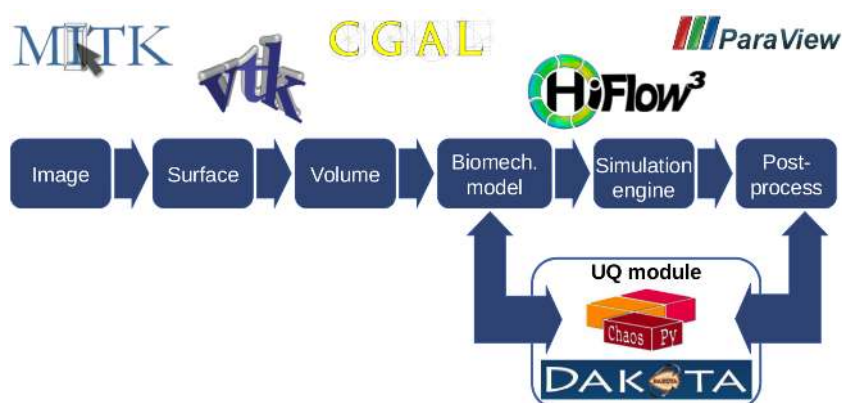


Figure 6.16: Schematic representation of an abstract biomechanical modeling workflow as in Figure 6.1, extended through the consideration and integration of UQ for the setup of the biomechanical model and simulation scenario via the pre- and post-processing components, also pointing out two established UQ toolkits, ChaosPy and Dakota.

MSML goes Semantics: Despite the fact that the MSML can enormously simplify the biomechanical modeling and simulation workflow as detailed in our work [SSS⁺14], the setup of meaningful biomechanical simulation scenarios still requires an in-depth numerics and simulation expert knowledge [SSSVH16]. On the one hand, a suitable biomechanical model has to be composed in order for the proper simulation of the elastic behavior of the concerned anatomical objects and of the physiological or surgical processes. On the other hand, numerical expertise is required for an optimal choice of numerical algorithms and discretization techniques and for the appropriate selection of the respective FE solvers for each problem.

In our follow-up work [SSSVH16], we describe an idea to circumvent these issues: We propose the concept and prototypic implementation of a *semantic surgery simulation system*, which encompasses the numerics knowledge and experience of modeling and simulation engineers. Via a structured semantic representation of numerics and simulation properties (in the framework of a specific numerics domain ontology) and via an underlying numerics properties decision tree, which is linked to a corresponding semantic representation of anatomical and surgical properties (in the framework of a surgical domain ontology), this cognitive system has the potential to properly deduce simple, patient-specific simulation scenarios. See Figure 6.17 for an illustration of a basic surgical sample scenario for liver surgery.

This being said, enhancing a framework like the MSML with ontologies in order to represent numerical modeling and simulation knowledge may well be the next big step towards *intuitive biomechanical surgery simulations* [SSSVH16]. Also, such a system will certainly improve the in-OR-usability and acceptance of surgery simulations in surgery assistance systems, and it can help make complex simulation technology available for non-simulation-experts.

Concluding, in this chapter, we have presented the MSML-based description of biomechanical modeling workflows which aim at the facilitation of the setup of patient-specific biomechanical simulation scenarios for surgery assistance. By the example of MVR surgery, we have shown

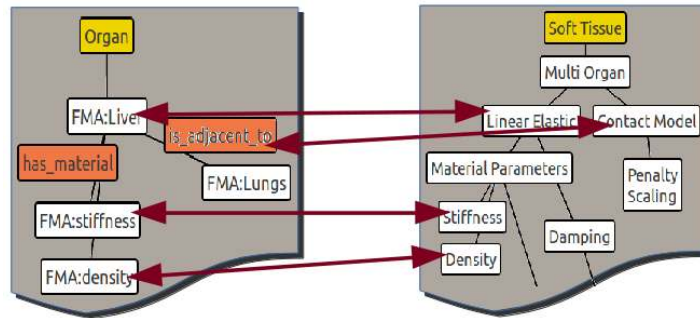


Figure 6.17: Sample coupling of two simplified property and dependency structures: By the example of liver surgery, anatomical and surgical properties/relations (depicted on the left side) couple with numerics and modeling properties/relations (on the right side).

how the MSML can act as a middleware between all tools used in the biomechanical modeling pipeline and how it simplifies the construction of biomechanical models for surgery simulation from patient-specific medical data such as 3D images and parameter data.

In the next chapter, we will explain how – on the basis of *cognitive software tools* and an *integrative, semantic system and data architecture* – we have our MSML-based MVR pre- and post-processing workflows and our HiFlow³-based MVR simulation application controlled in a way which is patient-specific on the one hand, and which is based on surgical expert knowledge on the other hand. Also, we will see how our cognitive MVR surgery assistance system is capable of processing heterogeneous medical data, and how it facilitates the combination and execution of all steps in the MVR surgery assistance workflow fully automatically by means of its data-driven software architecture.

7 Cognition-Guidance for Simulation-enhanced Cardiac Surgery Assistance

Preceding, we have detailed the conceptualization, implementation and application of our HiFlow³-based biomechanical MVR surgery simulation (Chapters 4 and 5) and of the MSML-based comprehensive, patient-specific MVR simulation preprocessing workflow (Chapter 6).

In this chapter, we will explain how – based on a common *semantic system and data architecture* and through *cognitive software components* – we manage and control these two in a way such that the resulting MVR surgery simulation scenarios are both patient-specific and based on surgical expert knowledge, and hold for integration into a *cognition-guided, simulation-enhanced patient-individual MVR surgery assistance system*. Therefore, we will present a cognitive tools pipeline, which is capable of processing heterogeneous medical data and surgical knowledge, and which allows to set up comprehensive, situation-adapted, patient-specific surgery simulation scenarios fully automatically.

First, in Section 7.1, we give a rationale for *surgical data science* and outline the basics of *clinical information processing* with focus on surgical knowledge representation and extraction. Following, in Section 7.2, we present our common *cognitive system architecture* along with the underlying *semantic data infrastructure* and the concept of *cognitive software components*. In Section 7.3, we will look at the specific case of MVR surgery assistance again, and show how by utilization of the before-mentioned concepts and architectures we set up a pipeline of *cognitive tools for simulation-enhanced cardiac surgery assistance*. Finally, in Section 7.4, we discuss the obtained results and give an *outlook* towards semantic surgery simulation.

7.1 Basics of Clinical Information Processing and Surgical Data Science

Holistic surgical treatment planning. For the planning of a *holistic* surgical treatment strategy, surgeons make use of their surgical expert knowledge and ideally account for all available patient information in order to deduce the best suitable surgical treatment strategy [MSD⁺09].

The decision for a potentially optimal surgical treatment for a respective patient, however, depends on several impact factors [MHW⁺15]:

- a large amount of heterogeneous *medical patient data*, which includes, among others, a patient’s personal information (age, sex, etc.), imaging results (e.g., from ultrasound or CT), health history records (on medication, previous diagnoses, treatments and surgeries), medical or clinical parameters (from laboratories, radiology, anesthesia, etc.), and generally anything else that is recorded before, during or after the surgical treatment;
- the technical development in *diagnostic or therapeutic procedures*, which also entails a

(possibly missing) required experience with the handling and operation of the respective OR technologies;

- the constantly growing *knowledge and evidence from clinical studies and literature*, which – in sum – constitutes the surgeon’s expert knowledge and surgical experience, i.e., on the one hand, factual knowledge from literature, scientific/medical/clinical studies or guidelines, and, on the other hand, practical knowledge from experience and expertise in surgical procedures;

Surgeon have to properly analyze and interpret these diverse pieces of information based on their individual knowledge and experience, and thereon-based make comprehensive decisions to derive the potentially best surgical treatment choice [MSD⁺09].

However, a surgeon often simply cannot consider all potentially available information and knowledge due to time constraints or due to a lack of interpretation experience. Also, following the renowned cognitive-psychological study of Miller [Mil56], the number of objects an average human can hold in his/her working memory (i.e., the human working memory capacity) is limited to 7 ± 2 . This, of course, also applies to human surgeons and to the number of studies or chunks of pieces of patient information they can consider and actively include in the planning of their surgical strategy. Besides that, surgical success is often hampered due to the fact that various surgical techniques exist or partly are still developed [BMD13], and by virtue of the common situation that there even are partly conflicting opinions on best surgical practices, which, however, are often chosen and applied without clear scientific evidence, but rather according to the personal experience of the operating surgeon.

With a constantly growing amount of medical data, guidelines and studies, recalling this vast amount of information in the right situation and at the right moment poses an ever-growing challenge, which should be *supported* by adequate technology.

How to make knowledge accessible to technical surgery support systems? Hence, for a corresponding *technical surgery support system*, the question one has to ask is: How can technology (i.e., a machine) *access* the entirety of this surgical knowledge along with the above-mentioned large amounts of patient data, *interpret* it, and conduct surgically meaningful reasoning on the data, in order to assist surgeons in their work, and eventually infer a possibly best surgical treatment strategy?

Since we are dealing with surgery simulations, along with this question one immediately finds that one has to tackle the challenging problem of managing, administering and maintaining a *surgical knowledge data base*, and that one has to implement a series of *cognitive algorithms* which are capable of intelligently working on the contained data (i.e., analyze and interpret the data under consideration of the represented surgical knowledge), and of processing this data for the purpose of suitable surgery simulation scenarios.

In the context of the state-of-the-art of cardiac surgery assistance, we have already depicted the two main strings of *medical information processing* for simulation-enhanced surgery assistance in Figure 3.1: the processing of medical image data and the processing of non-image data, which jointly stand for the entire set of surgically relevant machine-processed pieces of patient information. Besides patient data, the surgical assistance system is required to also have access

to a machine-usable representation of the above-mentioned surgical expert knowledge and experience (i.e., the factual and practical knowledge). Such *knowledge representation* is what actually will allow the technical surgery assistance system to properly analyze, process and interpret the given patient data, and to thereon-based infer further actions, such as to deduce a possibly optimal surgical treatment strategy, or to set up a meaningful, situation-adapted, knowledge-based, patient-individual biomechanical simulation scenario [SEZ⁺15, SPW⁺16, FMK⁺16].

A number of important technical developments in the fields of Data Science (DS), Machine Learning (ML) and Artificial Intelligence (AI) have contributed to the impressive series of accomplishments that have in the meanwhile been achieved in order to satisfy the above mentioned requirements. They have lead to an overall progressing digitalization of the OR, and gain increasing significance in today's surgery assistance systems, especially with respect to intelligent and holistic analysis and processing of medical information [BBF⁺15].

In the following, we will present the most prominent developments and introduce some of their basic features and functionalities in order for a better understanding of the technical details in the subsequent Sections 7.2 and 7.3.

Knowledge representation/extraction and reasoning. Most importantly, we will consider two fields of AI, knowledge representation and reasoning and knowledge extraction, which, on the one hand, aim at capturing and representing information about the world in a form that a computer system can utilize in order to reason and to solve sophisticated tasks, and on the other hand, provide methods for the extraction and creation of knowledge from general data sources [Bra85].

Several mechanisms and formalisms such as *semantic nets*, *frame languages* or *ontologies* have been conceptualized and implemented in the context of or in order to drive knowledge representation, e.g., for the application in *expert systems*. Associated with knowledge representation and naturally accompanying it, an impressive number of *automated reasoning engines* have been developed, since some of the main purposes of explicit knowledge representation are to be able to conduct reasoning about the knowledge (using classifiers), to make inferences (via inference engines), assert new knowledge, etc. Today, such reasoning components are embedded or integrated in a range of intelligent devices and systems [HRWL83] and help automate processes, e.g., in the surgical domain.

With increasing numbers of intelligent technologies, knowledge-based systems (and along with them the utilized knowledge representation mechanisms and the connected knowledge bases) have excessively grown over the last years. This gave rise to the development of so-called *ontologies* and the discipline of ontology engineering.

Ontologies. Ontologies are used for the formal declaration (i.e., naming and definition) of individuals, things, types, properties, attributes or relations between things and entities of a particular domain of discourse. Common components of ontologies, however, are not limited to these items, but may also include rules, restrictions, axioms or events.

With these capacities they limit the formal description complexity of sets of entities, concepts and their interrelationships, and thus perfectly fit the need of knowledge representation mechanisms [OGS09]. Through facilitating the logical and machine-interpretable organization

of information, they moreover essentially contribute to machines being able to properly capture and efficiently manage knowledge, which is key to their application to problem solving (e.g., via *Semantic Reasoning*) and hence to building powerful *cognitive AI systems*.

In the course of the years, a number of *ontology languages* (e.g., LISP, logic; CycL, KIF, OWL, RDF, etc.) have been developed, however, the distinguishing feature and capacity of an ontology is not the language in which it is written down, but the content, i.e., the set of *concepts* and *relations* that it provides as a way of thinking about the world and hence for formally describing it. We emphasize at this point that the latter is the central idea of ontologies, as it facilitates reasoning to be conducted, *not* on instances, *but* on concepts and relations.

In this respect, *modularity*, i.e., the ability to formally define boundaries around particular domains and specific problem spaces, is essential for these languages. In its modular setup, an ontology language corresponds to the necessary modular structure and setup of knowledge bases [RN10]. Given an arbitrarily complex world and many differing, competing views of this world, modularity is a substantial requirement in order for enabling the formal description of many specific, self-contained domains (using so-called *domain ontologies*) that are still able to communicate and integrate with each other (via so-called *upper or foundation ontologies*).

A *domain ontology* formally describes and represents things, concepts and relations which belong to or are an element of a specific part of the world. In doing so, they commonly are very specific and hence often incompatible. Nevertheless, as systems that rely on domain ontologies will expand at some point, it is then – as part of the overall ontology design – required to merge the respective domain ontologies into a more general representation. At present, merging ontologies that are not developed from a common upper or foundation ontology is a mainly manual piece of work. In contrast, domain ontologies can be merged automatically, if they derive from and use the same foundation ontology in order to provide a set of basic elements with which the meanings of the domain ontology elements can be specified.

An *upper or foundation ontology* correspondingly is a model or representation of very common concepts and objects that are generally applicable across multiple knowledge domains and hence across a wide range of domain ontologies. It therefore usually employs a core glossary that contains the terms and associated object descriptions as they are used in various relevant domain sets [ASS15]. As described above, an important function of an upper or foundation ontology is the support of a broad *semantic interoperability* among a large number of lower-ranked, possibly derived domain-specific ontologies. The lower ranking of a domain-specific ontology hence suggests that its concepts may be derived from or abstracted to a hierarchically higher-ranked upper ontology.

Some important examples for specific domain ontologies in the medical and clinical context are given by the FMA¹ [RM07], an ontology for the human anatomy, LOINC² [CMES10], an ontology for medical laboratory observations, and SNOMED CT³ [Rec08], a clinical terminology.

Essential, standardized upper ontologies available for use would include the BFO⁴ [ASS15] in the biomedical context, DOLCE [Rob11] for linguistic and cognitive engineering, Dublin

¹<http://sig.biostr.washington.edu/projects/fm/>

²<http://loinc.org/international>

³<http://www.ihtsdo.org/snomed-ct>

⁴<http://ifomis.uni-saarland.de/bfo/>

Core [Dub13] to describe web resources, and UFO [GHW03, GW10] to analyze general structural and conceptual modeling constructs. In our work, we put particular emphasis on the BFO, the *Basic Formal Ontology*, which has come to be established and adopted as a foundational ontology by many prestigious projects in the area of biomedical informatics [TRDB10, RCB⁺07].

To give ontologies a substantial formal basis, large collaborative research and development projects have recently integrated concepts from knowledge representation and reasoning (namely frame languages and automated reasoning components) with markup languages that are based on XML.

The *Resource Description Framework* (RDF) [HPS14] is a standard model for data interchange on the Web. Corresponding to the central idea of ontologies, it provides basic capabilities to define knowledge-based objects by means of fundamental features such as *object properties* or *subsumption relations*. The common RDF data model is based on *triples* (consisting of subject, predicate and object), and it extends the Web linking structure (i.e. the use of uniform resource identifiers, so-called URIs) in order to name the components of the triple, i.e., the relationship between things on the one hand, and the two ends of this link on the other hand. Using this simple data model, RDF allows structured and semi-structured data to be mixed, exposed, and shared across different applications. From an abstracted point of view, the respective linking structure forms a directed labeled graph, where the edges represent the named links between two resources which in turn are represented by the graph nodes [MACP02]. Other technologies, like *OWL* or *SKOS*, build on RDF and provide languages for defining structured, Web-based ontologies [HPS14].

The *Web Ontology Language* (OWL) is a Semantic Web language designed to represent complex knowledge about things, sets of things, and relations between things [HKP⁺12]. It is a computational logic-based language, such that knowledge expressed in OWL can be exploited by computer programs, e.g. using classification engines [BLHL01]. This makes it a key technology in the context of intelligent systems. OWL documents are often referred to as *ontologies* themselves, too. An important feature of such ontologies is that they can be published in the Web and that they may refer to or be referred from other OWL ontologies.

Both, OWL and RDF are part of the W3C's Semantic Web technology stack, see Figure 7.1 or: <https://www.w3.org/standards/semanticweb/>

Linked Data. When objects in the above RDF or OWL data models are linked or referred to/from, this is done in consistence with the principles of *Linked Data* [BL06]. The term *Linked Data* was coined by Tim Berners-Lee in a 2006 design note about the *Semantic Web* project.

Linked Data is a method of publishing structured data on the Web. It therefore builds upon standard Web technologies such as URIs for resource identification, RDF for abstract conceptual semantic annotation, and HTTP for communication and data exchange. However, the main purpose of Linked Data is not to be useful for human readers, but rather it allows to formally describe and share information in a way that can be read and interpreted automatically by machines and computers. In allowing data from arbitrary sources to be interlinked and connected, it also facilitates data to become more useful through semantic queries, e.g. via the *SPARQL Protocol and RDF Query Language (SPARQL)* [BHBL09].

The four *principles of Linked Data* (published in Tim Berners-Lee's *Linked Data* note of 2006 [BL06]) paraphrasedly state the following:

- Use URIs to name and identify things;
- Use HTTP URIs so that these things can be looked up, interpreted and dereferenced;
- Provide useful information about what a name identifies when it is looked up, using open standards such as RDF (and its serializations like RDF/XML, N3, etc.), SPARQL (or similar/equivalent query languages like N3QL, or rule languages like SWRL, etc.);
- Refer to other things using their HTTP URI-based identifiers when publishing data on the Web;

The concept of Linked Data thus makes the *Semantic Web* browsable as every object and every type of relation is *linked* as part of the so-called ***Giant Global Graph***.

Semantic Web. As of 2001, projects associated with the *Semantic Web* represent one of the most active research fields in the area of knowledge representation and reasoning [BLHL01]. *Semantics* is the formal, mathematical study of the meaning of (programming and description) languages (and of data or algorithms written in that language). It is conducted through evaluation of the meaning of syntactically legal strings (that are defined in terms of a respectively specific language) and by analyzing the subsequent computation that is triggered or involved.

The *Semantic Web* thus extends the standard Web by means of a layer of semantics on top of the current Internet layer, in which information is given a well-defined *meaning*. Also known as the *Web of Data*, it enables machines to *read, interpret and process* data and knowledge using inference and deductive reasoning techniques.

In order to facilitate this idea, the Semantic Web builds on the principles of *Linked Data* for publishing structured data, and thus allows data to be interconnected, which also makes them usable through semantic queries [BL06]. Moreover, through utilization of the common *RDF data model* and a standardized *ontology language*, data and even implicit knowledge worldwide can be integrated [OGS09].

The realization and implementation of the Semantic Web is based on a collection of ***technologies*** that are specified as *standards* by the *World Wide Web Consortium* (W3C) [BL06]. These technologies include:

- *Uniform Resource Identifiers (URIs)*, an unambiguous string of characters used to identify or refer to a resource in the Web; the URI may or may not be resolvable on the Web; resolvable URIs are also called URLs (Uniform Resource Locators) or Web addresses;
- *Resource Description Framework (RDF)*, a general method and model for describing information/data;
- *RDF Schema (RDFS)*, for the declaration of the vocabulary which is used in RDF;
- *Simple Knowledge Organization System (SKOS)*, which is designed for representation of classification schemes, taxonomies or any other type of structured controlled vocabulary;
- *SPARQL*, an RDF query language and protocol for RDF-based data;
- *Notation3 (N3)*, a shorthand non-XML serialization of RDF models that is designed with human-readability in mind;

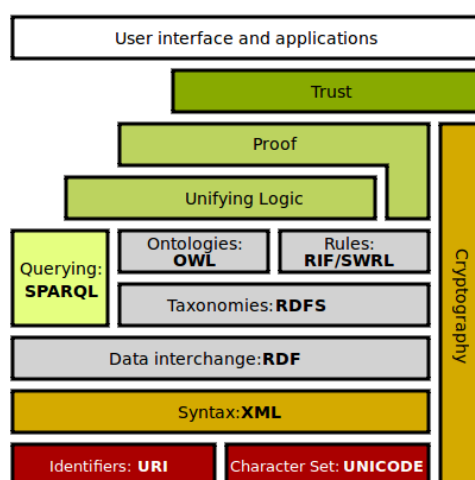


Figure 7.1: The Semantic Web Stack. [Illustration reprinted from Wikimedia Commons.]

- *N-Triples*, a line-based, plain text serialization format for RDF graphs, used as a format for storing and transmitting data;
- *Turtle* (Terse RDF Triple Language), which is an alternative format for expressing data in the RDF data model with a syntax similar to SPARQL;
- *Web Ontology Language (OWL)*, a family of knowledge representation languages that enable authoring and linking of ontologies;
- *Rule Interchange Format (RIF)*, a framework of web rule language dialects supporting rule interchange on the Web, e.g., between SWRL and OWL;

Using the respectively mentioned components and standards, the so-called **Semantic Web Stack** (see Figure 7.1) illustrates the setup and architecture of the Semantic Web.

Critics have long been questioning the feasibility of the Semantic Web. Above all, they see a number of perpetual challenges that have always created problems for automated reasoning systems. These include, among others, its vastness (e.g., in the context of duplicated terms, etc.), its vagueness (with respect to imprecise or ambiguous definitions of concepts/objects), uncertainties (when precise concepts have uncertain values), inconsistency (in case of logical contradictions in definitions and terms), or deceit (intentional misleadings in object/concept definitions).

However, proponents argue that applications in industry, biology, medicine and human sciences research have already proven the validity and usability of the original concept according to Tim Berners-Lee [BLHL01]. In 2013, more than four million Web domains contained Semantic Web markup [V.G13]. In particular, in the fields of bioinformatics, life sciences and health care, tools from the Semantic Web have in the meanwhile been successfully applied for years in order to support data integration [VH01] and in translational research [RCB⁺07]. For an overview, we refer to [CPWS09].

For more general information on the Semantic Web, we refer to the W3C's Semantic Web technology stack, see: <https://www.w3.org/standards/semanticweb/>

Driven through ongoing research and development activities in the above introduced context, and based on the intelligent concepts and techniques which thus came within reach, recent years have shown a number of sophisticated advances thrive in the domain of biomedical informatics and in surgery assistance; again, for more information, we refer back to Section 3.1.4.

In this work, using the above introduced technologies and standards, we particularly aim at conceptualizing and setting up a *cognitive simulation-enhanced cardiac surgery assistance system architecture*. This cognitive system architecture will include and enable the execution of a *cognitive tools pipeline* for knowledge-based, patient-specific, simulation-enhanced cardiac surgery assistance.

Therefore, we will show below, how – under consideration of semantically represented surgical expert knowledge – we manage to patient-specifically control our HiFlow³ MVR simulation (as presented in Chapters 4 and 5) and the MSML-based MVR simulation pre- and post-processing workflows (as in Chapter 6). We will give details on how and why the conceptualized cognitive cardiac surgery assistance system is capable of processing heterogeneous medical data, and how it facilitates the combination and execution of all steps in the MVR surgery assistance workflow fully automatically by means of its data-driven software architecture.

With our work, we hence aim at making a step towards the integration of a cognitive, situation-adapted, knowledge-based, patient-individual biomechanical MVR surgery simulation into a cardiac surgery assistance system in order to thus progress towards a more comprehensive simulation-enhanced surgical treatment planning.

7.2 Semantic System Architecture for Cognition-Guided Surgery Assistance

In the previous Section 7.1, we have outlined the most important technical developments in the context of Surgical Data Science, especially with respect to knowledge and information processing in surgery assistance and in clinical information systems.

Going further, in this section, in 7.2.1, we will present a concrete system architecture for a *cognitive surgery assistance system*, and particularly point out the underlying *computing and data infrastructure* along with its common *knowledge base*. The respective conceptualization and implementation results from a joint research effort with the German Cancer Research Center (DKFZ Heidelberg) and the Karlsruhe Institute of Technology (KIT) in the framework of the collaborative research center *Cognition-Guided Surgery* (SFB TRR 125).

In 7.2.2, we outline how arbitrary *software components*, that are part of a surgery assistance workflow, can be wrapped semantically, and how the thus resulting *Cognitive Applications* are then enabled to intelligently process semantically annotated medical patient data and information fully automatically in a surgery assistance pipeline.

We thus want to give the foundation for our *cognitive, simulation-enhanced MVR surgery assistance setup*, which we will then present as an exemplary application in the next Section 7.3.

7.2.1 Semantic Data Infrastructure and Clinical Knowledge Representation

The vision of the interdisciplinary, multi-institutional collaborative research center *Cognition-Guided Surgery* (SFB TRR 125) is a *cognitive surgery assistance system*, which – as explained

in detail in the introduction in Chapter 1 – is capable of 1) perceiving pre-, intra- and post-operative data, of 2) analyzing and interpreting this information with the help of 3) a common surgical knowledge base, and of 4) then inferring a context-sensitive action; compare again Figure 1.3.

Using the underlying comprehensive *SFB knowledge base*, highly specific, dedicated algorithms and applications are to utilize the medical data and derive additional information, such as 3D geometric meshes of organs [SSS⁺14], treatment recommendations [MHW⁺15], or biomechanical surgery simulations [SEZ⁺15]. In order to realize and implement such a system and to make it accessible to surgeons, a *coherent representation of the surgical and scientific knowledge* and a *unified access* for miscellaneous *information processing algorithms* to the *entire heterogeneous medical data set* is imperatively required.

Therefore, in a joint work [FMK⁺16], a concrete system architecture has been presented to hold as a basis for a *cognitive surgery assistance system*. We particularly pointed out the underlying *semantic data infrastructure*, which integrates the above-mentioned highly diverse but valuable pieces of information by using a common knowledge representation.

A unified access to the infrastructure via a common semantic interface and a common technical platform facilitates the development and adaption of algorithms and applications that analyze and process the data, in order to infer, e.g., biomechanical surgery simulation scenarios, which may support surgeons in finding the best possible therapy for individual patients.

In order for a *common representation* of the respective surgical *knowledge and information*, and for the *integration* of the enormous amount of heterogeneous medical or technical *data* into a collective system on which semantic information processing can be performed, this system has to satisfy – at least – the following requirements:

- *Data storage and access*: Standardized file types for storing specific clinical and scientific data must be well-defined, and the data must be accessible in a uniform way via flexible programming interfaces such that communication and interchange of data between clinical information systems and OR workstations is facilitated.
- *Data integration*: The system's entire data and information, i.e., both clinical and scientific data as well as surgical guidelines and best practices, must be described and linked semantically using a common knowledge representation.
- *Semantic data query*: Comprehensive semantic queries on the entire data set should be enabled in order to identify and allow access to relevant subsets of the data, e.g., for setting up adequate surgery simulation scenarios, or for case-based reasoning.

For more details on the requirements list, e.g., with respect to patient data safety and protection, or to the deployment of the respective infrastructure, please refer to our common work [FMK⁺16].

In the following, the concept and implementation of our joint semantic system architecture and its underlying data infrastructure will be presented. It is designed to meet the above requirements and to thus provide access for general information processing algorithms to the semantically annotated medical data set and to surgical knowledge through standardized interfaces. Figure 7.2 schematically illustrates the entire knowledge base architecture along with

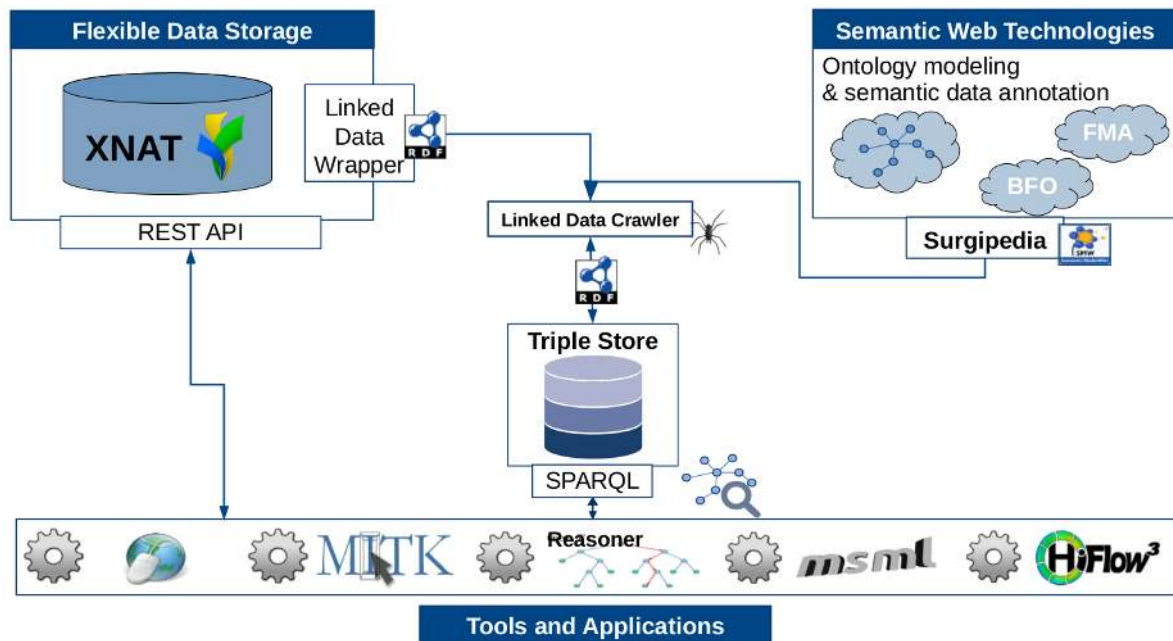


Figure 7.2: Schematic illustration of the central components of our *semantic system architecture* with focus on the underlying *data infrastructure* of our *knowledge base*, which combines a flexible data storage (XNAT) with Semantic Web Technologies and with standardized application interfaces:

As a **flexible data storage**, the *Extensible Neuroimaging Archive Toolkit* (**XNAT**) provides miscellaneous, competent interfaces for various **tools and applications**. **Semantic Web technologies** facilitate, via *Surgipedia*, a *Semantic MediaWiki* (SMW) instance, the collaborative semantic modeling of data and the linking of external *ontologies*, such as the *BFO* or the *FMA*. A **Linked Data Crawler** aggregates the interlinked semantic information stored in these two systems into a central **Triple Store**. This triple store then contains the system's entire information and hence allows for semantic queries via a *SPARQL Protocol and RDF Query Language* (*SPARQL*)-endpoint. As such, the Triple Store serves as a unified semantic interface to the system's entire data set. For more information, see the joint work [FMK⁺16].

its specific components. Subsequently, we will give further details on these components and on how they meet the identified requirements.

Flexible semantic patient data storage and access. In the framework of the collaborative research center *Cognition-Guided Surgery*, in order to satisfy the requirement for *flexible semantic patient data storage and access*, it was chosen to make use of the *Extensible Neuroimaging Archive Toolkit (XNAT)* [MORB07]. XNAT is an extensible open-source data management platform and focuses on storage of medical images, but is also capable of storing arbitrary non-image data.

As the central data storage system, XNAT administrates data based on a dedicated *XNAT data model*, which is composed of three core data types: the *project* (to identify a specific study, examination or surgery), the *subject* (to identify the respective (real) person/patient undergoing medical/clinical examination/surgery) and the *experiment* (to identify the event by means of which medical/clinical data were recorded, e.g., an ultrasound or CT session, an interrogation, or surgery simulation results). The full XNAT data model is defined in an XML file, which is accessible via: <https://central.xnat.org/schemas/xnat/xnat.xsd>

In order to access medical or clinical data, XNAT provides versatile interfaces for various tools and applications: its comprehensive REST⁵-conform interface (*REST API*⁶) and an integrated *DICOM*⁷ gateway provide a range of options for an integration into clinical and scientific setups. Utilizing these interfaces, XNAT can be accessed (e.g., in order to upload, download or to query data), for instance, via a *Web Browser*, a *Picture Archiving and Communication System (PACS)* or in a programmatic way via *command line tools*, via *Python*, *Java* or via the *MSML CLI*. Also, as will be outlined below and in the following subsection 7.2.2, data in XNAT can be accessed and manipulated by means of the MITK-based *Cognition-Guided Surgery Workbench* and via so-called *Cognitive Applications*.

With respect to *patient data protection*, XNAT provides means to make use of external software toolkits, such that it is guaranteed that authorized clinical users have full access to personal data, whereas researchers and engineers will only be granted access to de-identified and anonymized data, thus satisfying necessary features of a privacy-compliant research data infrastructure. For instance, between the hospital PACS and XNAT, DICOM data is automatically de-identified using the certified software *PID Generator*, published by the German telematics platform TMF [FP16], which also supports the re-use of pseudonyms for long-term data accumulation [PR04].

Semantic data integration. In order for a cognition-guided surgery assistant system to be able to perform semantic reasoning on medical data and knowledge (and to thus create new knowledge and insights, e.g., treatment recommendations or suitable surgery simulation scenarios), the respective information and stored data needs to be *linked and annotated semantically*.

⁵REST: Representational State Transfer, an architectural style for information systems on the Web, which, among others, ensures some of the Web's key features such as extreme scalability and robustness to change.

⁶REST API: An API implemented using HTTP and adhering to the principles of REST to allow actions on the Web, most commonly to create, retrieve, update or delete resources through commands like POST, GET, PUT or DELETE.

⁷DICOM: Digital Imaging and Communications in Medicine, an open standard for handling, storing, printing and transmitting information in medical imaging.

For this purpose, different *data types* had to be surveyed, which would eventually emerge within the overall system of our collaborative research center *Cognition-Guided Surgery* and in the context of its information processing pipelines. Thereby, in a common effort, we also considered data types arising from our simulation-based MVR surgery assistance pipeline, and respectively complemented a list of supported *data formats* in accordance with our simulation requirements. In this context, for the definition of the data formats, standardized and well-established file formats with existing open-source implementations (e.g., DICOM for general image data, VTK for structured/unstructured geometry data arising from segmentations or simulations, XML, JSON, RDF, TTL and OWL for document-based data and semantic annotations) were preferred over others in order to thus ensure compatibility and interoperability between multiple systems. For more details, we refer to our joint work [FMK⁺16].

Thereon-based, we jointly defined a common *data ontology*, which semantically describes all data types and formats that are to be used within our collaborative research center *Cognition-Guided Surgery*. In the process of defining and implementing this data ontology, we stick to and derive from the formal concepts of the *Basic Formal Ontology (BFO)* [ASS15], which – as was mentioned in the previous Section 7.1 – is an established upper/foundation ontology. Upper ontologies constitute a key technology for accomplishing the *semantic integration* of heterogeneous information from miscellaneous sources [TRDB10].

Moreover, in order to ensure high *interoperability* with existing technologies as well as easy availability of tools, we chose to rely on established W3C-specified standards from the Semantic Web, namely the *Resource Description Framework (RDF)* and the *Web Ontology Language (OWL)*, both of which follow the principles of *Linked Data*; see again Section 7.1.

In order to then allow for the collaborative modeling of semantic annotations (meta information) for all knowledge-base-relevant data instances, *Surgipedia*, a *Semantic MediaWiki* instance [KVV06], was deployed. A Semantic MediaWiki (SMW) is an extension to MediaWiki (an open-source wiki application) which allows for annotating semantic data within wiki pages, thus turning a wiki that incorporates this extension into a *semantic* wiki. While traditional wikis commonly contain only text, which computers can neither understand nor evaluate, SMW adds semantic annotations that allow a wiki to function as a collaborative database. Hence, data that has been encoded can be used in semantic searches, for the aggregation of pages, displayed in formats like maps or graphs, and exported to the outside world via formats like RDF. In addition to its application for the modeling of semantic annotations, Surgipedia is utilized for integrating with and linking to additional external terminologies and ontologies, such as the *Foundational Model of Anatomy (FMA)* or *SNOMED CT*, and thus for supporting a broad *semantic interoperability* of our collaborative data ontology with other domain ontologies.

As such, Surgipedia eventually allows to link heterogeneous medical and clinical data in XNAT with medical terms and surgical knowledge that is defined, modeled and represented through the respective ontologies; see Figure 7.3.

In order to standardize and automate the process of data annotation, *data annotation templates* were introduced to Surgipedia. These templates define for each data type which information needs to be included (either necessarily or optionally) into the respective annotation. It is thus ensured that the individual annotation also accords with the underlying data ontology. Resulting annotations are then stored as RDF files, and typical information contained in these RDF files could be: file format (e.g., VTK or DICOM), file content (e.g., blood measurements,

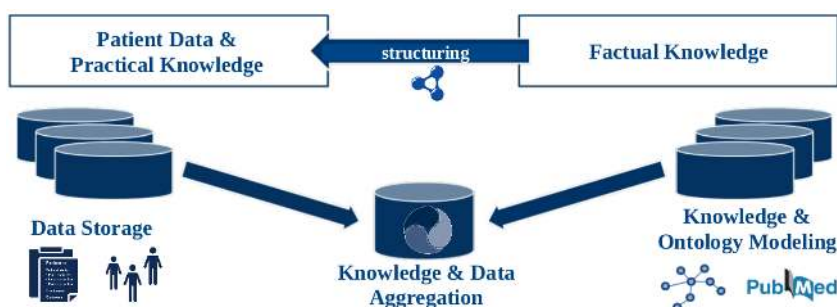


Figure 7.3: Illustration of the *knowledge base* setup in the context of our common semantic system architecture: The system setup facilitates that heterogeneous patient data and practical surgery knowledge can be aggregated and linked to formalized and semantically represented factual knowledge. Dedicated mechanisms and algorithms for data processing and data integration then allow a unified access to the entire information, e.g., in order to enable semantic reasoning in the context of simulation-enhanced assistance during cardiac surgery.

imaged anatomical structures, simulation results), file source (e.g., a specific algorithm, tool or medical device) or links to related annotations or data files. New data annotation templates can easily be created in Surgipedia by using a *data annotation template creator*, which itself is based on a semantic form that helps the user to properly create the template, e.g., through autosuggestions or autocompletions according to the data ontology. Figure 7.4 shows the exemplary procedure of creating a such data annotation template for MSML-generated HiFlow³-compatible MVR simulation scenario setup files (as described in Chapter 6) using the Surgipedia data annotation template creator.

The actual semantic annotation of the data in XNAT is then achieved through completing the respectively adequate data annotation template with the concrete data information, i.e., for example, the XNAT URL of the related file, the patient ID, the morphology information for the respective patient, or the specific blood pressure values. Lastly, the annotation file is stored in XNAT at a file location similar to the original described data file's location. Following this procedure, every data instance in XNAT is to obtain a semantic annotation, i.e., one (or more) attached RDF annotation file(s).

Semantic data queries. The entire information, including the semantic content of Surgipedia as well as the RDF annotations in XNAT, is aggregated by a *Linked Data Crawler* [IUBH10], a Web Crawler that is able to traverse and consume the Web of Linked Data, into a **Triple Store**. This Triple Store (or, in the SFB, rather: the endpoint to an installation of the *OpenVirtuoso triple store*⁸ software) offers querying capabilities over information stored in the SFB knowledge base, and hence represents a data warehouse that is specifically designed to handle *semantic data*; see again Figure 7.3.

For XNAT, a *Linked Data Wrapper* has therefore been implemented to convert all XNAT data into RDF, thus linking raw data to corresponding ontological concepts and dereferencing

⁸https://www.w3.org/2001/sw/wiki/OpenLink_Virtuoso/

Special page

Annotation Template Creator

Create/Edit Template

Template ID:

Label:

Data Information:

Information Content Quality: +

Information Representation:

Contact Person:

Is About: +

Has Source:

Project:

Figure 7.4: Exemplary utilization of the *Surgipedia data annotation template creator* for the creation of the data annotation template for MSML-generated HiFlow³-compatible MVR simulation scenario setup files.

URIs. It hence serves as an entry point for the Crawler for parsing the XNAT project structure along with the contained RDF files. Thereby, instead of transferring the raw data files, only the respective XNAT links and further meta data descriptions are added to the Triple Store.

As a consequence, only data files for which an RDF annotation file is given (i.e., stored in XNAT at a location similar to the actual, described data file's location) are considered by the Crawler, and hence accessible, interpretable and processable through the system. Also, nothing but the information which is semantically described in the RDF annotation file is detectable and available for the knowledge base and for the entire cognitive assistance system.

As the result of the crawling process, *RDF graphs* (i.e., complex, abstract triple structures that represent sets of objects along with their inter-object relations) from different sources are created and aggregated into the central Triple Store. The entire semantic content of the Triple Store is then accessible, e.g., on an *HTML webpage*, which contains general information on triples and ontologies, through a *Facet Browser*, a tool/GUI for browsing data by means of search terms, labels and entity URIs, or via the *SPARQL endpoint* for semantically querying the data. The *SPARQL Protocol and RDF Query Language (SPARQL)*-interface of the Triple Store allows for comprehensive semantic operations (such as aggregations, sub-queries, negations, constraining, value testing, etc.) to be executed on the entire data set in order to, e.g., perform semantic reasoning, or to carry out semantic search queries, for instance, through *Cognitive Applications* (which will be presented in the next subsection 7.2.2).

The Triple Store thus serves as a *unified semantic interface* to different data sources and storage systems. The overall RDF graph in which the entire data is stored can be found via: <http://wissensbasis.sfb125.de/>

Again, for more details, we refer to our original work [FMK⁺16].

Tools and Applications to access the Knowledge Base. Via the central Triple Store, a unified semantic interface to the overall information storage system is provided and gives access to the contained data for miscellaneous data processing tools and applications.

Amongst these tools and applications, we particularly want to mention the ***Cognition-Guided Surgery (CGS) Workbench***, which is based on the aforementioned *Medical Imaging Interaction Toolkit (MITK)* [NZS⁺13], and which unifies the different technical interfaces of the components of the above described semantic infrastructure into a common application platform.

Beyond its many technical features, for details on which we refer to [FMK⁺16], the CGS Workbench comes up with a user-friendly interface for surgeons and non-engineering staff to the proposed semantic infrastructure. For instance, the *MITK XNAT Treebrowser Plugin* provides a convenient way for browsing, uploading and downloading any kind of data from XNAT. Using this plugin, it is, e.g., possible to investigate on all relevant data created and consumed in the biomechanical modeling and simulation workflow (from 3D images, via segmentations, to biomechanical model representations and to simulation results) via a single GUI application, which the surgeon is already familiar with, and, moreover, is easy and intuitive to operate. A detailed overview and how-to on the different features, and both binary installers and the source code of the CGS Workbench are available at: <http://mitk.org/CGS>

Apart from the *CGS Workbench*, an important class of algorithms working on the data and information that is contained and represented in our semantic infrastructure is given by so-called *Cognitive Applications*, which we will have a closer look at in the following subsection 7.2.2.

Summarizing, based on a common semantic data and knowledge representation infrastructure and system architecture, and through adequate software tools and application interfaces, the proposed semantic system setup allows for *cognition-guidance* and meets the requirements of a *cognitive surgery assistance system*. Moreover, it has successfully been applied in use cases in liver [MHW⁺15] and heart [SPW⁺16] surgery assistance, which demonstrates the feasibility of the overall system setup.

7.2.2 Semantic Software Architecture and Cognitive Software Components

In addition to the above-presented *semantic data infrastructure*, in this subsection, we will outline the setup of our *semantic software architecture* as partly published in our work [SPW⁺16].

Architecturally built on top of the above-presented semantic data infrastructure, we put the following clinical application-inspired requirements on our semantic software architecture:

- *Tools integration:* A variety of medical data and information processing tools and algorithms need to be incorporated into *complex information processing pipelines*. These tools need to be able to run on different (possibly distributed) computer systems, from OR workstation to High-Performance Computing (HPC) cluster, depending on the respective algorithm's sensorial and computational requirements. Moreover, the respective tools need to be (remotely) executable from any workstation, be it in the OR or in another clinic department.
- *Data consolidation and enhancement:* In order to process all available information and heterogeneous data sources (images and non-imaging data), miscellaneous tools need to be able to communicate and to transform, consolidate and further process data in order to eventually provide enhanced information to surgeons and OR staff.

- *Automated information processing*: Data and information should – as soon as it is available for further processing – be processed automatically via self-composed, suitable information processing pipelines in order to thus relieve surgeons and OR staff as much as possible and to provide newly inferred information and comprehensive, useful results as quick as possible.

In order to meet these requirements our approach implements a pipeline of so-called *Cognitive Applications* which mutually benefit from and integrate into the *semantic data infrastructure* and the *common SFB knowledge base* concept presented in Subsection 7.2.1. Cognitive Applications were originally developed in the *Semantic Web* context [GMP⁺14, PMK⁺15], and are based on *Linked Data principles* [SSHS13].

Cognitive Applications. Cognitive Applications are, in essence, medical data interpretation and processing algorithms that are annotated with semantic meta data and wrapped behind a common RESTful Web interface [PMK⁺15].

The uniform RESTful Web interface provides flexibility in the algorithm’s remote accessibility, execution and result retrieval. The algorithm itself can be hosted on any server, and it makes its functionality accessible via standardized Web technologies – URIs for resource and endpoint identification, and HTTP for communication and data exchange, independently from its respectively underlying software architecture.

The semantic machine-readable service description, which is based on *Linked Data* and implemented in RDF, specifies exactly all necessary information to correctly employ and execute the algorithm. It comprises the specification of the algorithm’s operation, functionality, features and classification, and the definition of algorithm inputs/outputs and parameters. More precisely, the functionality of an algorithm is connected to a lifted, RDF-modeled representation of its required inputs (*preconditions*) and its produced outputs (*postconditions*); see Figure 7.5 for a schematic illustration. If provided with suitable *RDF-annotated* data, which matches the respective input description (*precondition*), this allows for running the Cognitive Application over the Web. Also, after execution, the results of a Cognitive Application not only comprise the original result data of the wrapped algorithm, but moreover include a corresponding RDF-annotated description of this data.

Hence, Cognitive Applications not only have a Linked Data description, but also consume and produce Linked Data, which in total – as will be detailed below – facilitates their communication and interaction with the SFB knowledge base via *Surgipedia*, the *Triple Store* and the *Linked Data Wrapper*.

The ontological annotation of data interpretation and processing algorithms brings a number of advantages: Algorithms can automatically be searched for based on their inputs/outputs (e.g., a 4D image of an MV, a clinical patient report, or simulation results) or classifications (i.e., for instance, medical imaging algorithms versus simulation tools). Moreover, we enable the flexible *composition* of miscellaneous algorithms into complete *data processing pipelines* (depending only on the respective algorithms’ pre/postconditions, i.e., on their input/output annotations). The individual steps in these pipelines may run on different computer systems: from OR workstations, for simple click-and-check processes, to HPC clusters, for computationally expensive tasks like numerical simulations; see Figure 7.6. Hence, Cognitive Application pipelines are able to process heterogeneous data by means of various single data processing tools

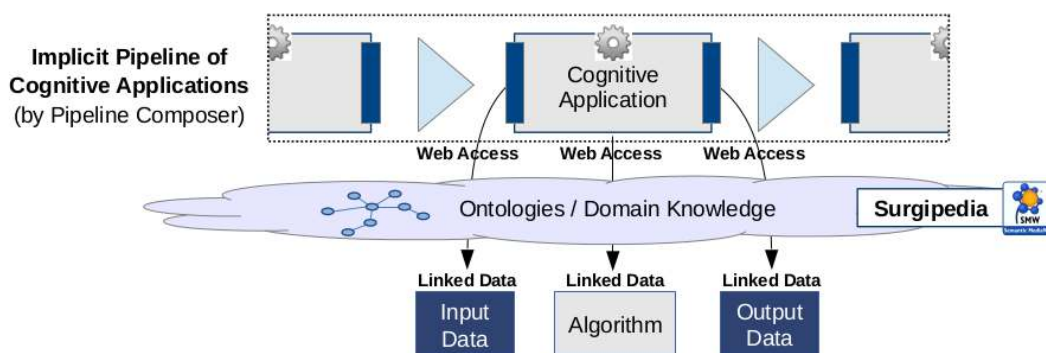


Figure 7.5: Illustration of the general functional setup and the interfaces of a *Cognitive Application* in the context of our common semantic data infrastructure. The declarative, cognitive nature of its semantic description is to be particularly emphasized.

that are all made accessible through semantically described Web APIs. Lastly, the composed pipelines can be executed *automatically*, provided that their individual component algorithms are fully-automated themselves, too. For a more detailed and Web-oriented introduction, see the works of Gemmeke [GMP⁺14] and Philipp [PMK⁺15].

Cognitive Applications in the context of the SFB Data Infrastructure. As mentioned above, our Cognitive Application-based software architecture is built on top of the *semantic data infrastructure* presented in the previous Subsection 7.2.1. Hence, according to the design of the central *SFB knowledge base*, the communication and interaction between knowledge base and Cognitive Applications is based on *Linked Data*, too. More precisely, in order to process medical patient data that are contained in the XNAT storage system, our Cognitive Applications only communicate with the *Linked Data* interface of XNAT, the *Linked Data Wrapper* – see again Figure 7.2 – , through which RDF-based XNAT file annotations are made accessible. This eventually facilitates the up- and download of original data files to/from XNAT through the respective Cognitive Application. Also, the thus provided up- and download functionality of the real resource instantiation between the application computer system and the XNAT storage system enables the potential further processing of data through potential subsequent Cognitive Applications in the respective pipeline that may actually run on a different compute server system – again, provided that the available data output (*postcondition*) from an application is accepted as input (*precondition*) of a respectively subsequent application. We refer back to Figure 7.6, which illustrates the mechanisms of communication and data exchange between Cognitive Applications, XNAT and application computer systems in the context of the common semantic data infrastructure.

At this point, it is to be emphasized again, that Cognitive Applications are not only semantically annotated with respect to their input/output data, but also with respect to their operation and functionality. The terms and relations used for describing the algorithms and their properties (i.e., for instance, their respective inputs/outputs, functionalities, classifications) have jointly been defined in the context of a common *algorithm ontology*, which – similarly to the above-mentioned *data ontology* – ensures a semantic consistency for the entirety

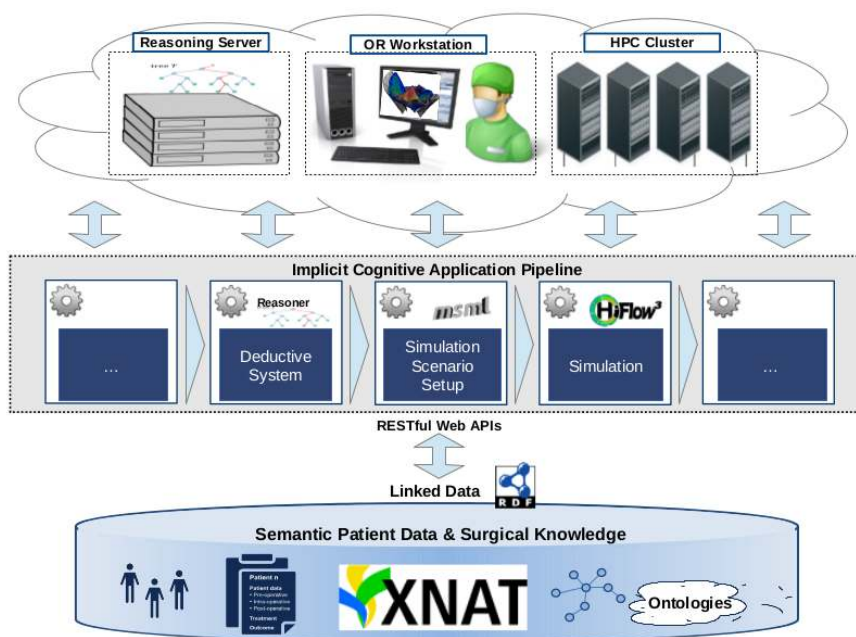


Figure 7.6: Exemplary *Cognitive Application* pipeline architecture for medical information processing, also illustrating the interaction with the semantic data infrastructure and the utilization of miscellaneous compute server systems.

of our SFB-internal tools and algorithms (all of which are to be usable in the overall surgery assistance system). Along with the data ontology, this algorithm ontology is part of the overall *Cognition-Guided Surgery (CGS) ontology*, see Figure 7.7. Eventually, it serves for the *semantic integration* of the developed software components, and for enabling communication between Cognitive Applications and the SFB knowledge base. As an example, it actually enables Cognitive Applications to match and access suitable RDF-annotated data in XNAT via the Linked Data Wrapper.

The set of medical assistance tools and data processing algorithms is itself also part of the common *SFB knowledge base*. Therefore, being semantically annotated with respect to the common CGS ontology by default, all tools and algorithms additionally get *registered* via *Surgipedia* so they are made discoverable.

Through utilization of a *Surgipedia algorithm annotation template*⁹, which assists in capturing the semantic algorithm annotations, we ensure that all information needed for the algorithm's invocation is properly described and compatible with the common data and algorithm ontologies, e.g., through allowing for the usage of prevalent input/output options. The respective template is similar to the *Surgipedia data annotation template* which was mentioned in the context of the above data ontology in Subsection 7.2.1. By filling this template, each Cognitive Application is simultaneously allotted a Semantic MediaWiki page on Surgipedia, which eventually facilitates the usage of Surgipedia as a *Cognitive Applications repository*.

Summarizing, Cognitive Applications are Linked Data consuming and producing Web ap-

⁹See: http://surgipedia.sfb125.de/wiki/Form:Cognitive_App_Semantic_Description

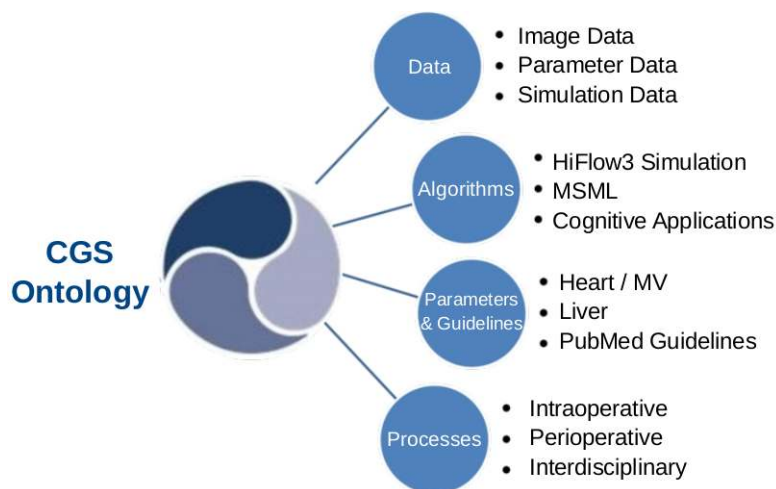


Figure 7.7: Illustration of the concept of the jointly set-up overall *Cognition-Guided Surgery (CGS) ontology*, which is composed of the *data ontology*, the *algorithm ontology*, a *patient parameter and surgical guidelines ontology* (see [MHW⁺15]) and a *surgical treatment process ontology* (see [KJW⁺15]).

plications, which seamlessly interact with the common semantic data infrastructure and the underlying knowledge base: They implement data-processing applications, which benefit from the knowledge base in using patient data and knowledge, and which also feed their results back into it. With Surgipedia being used as the data hub and core of the common knowledge base, we are able to combine all cognitive software components and to access and query the entirety of annotated data via the Triple Store and its SPARQL endpoint.

Automated pipeline composition and execution. In order to allow for the *automated composition and execution* of potentially complex *information processing pipelines* that consist of multiple Cognitive Applications, first, a *Pipeline Composer* needs to find and compose suitable sequences of Cognitive Applications into adequate pipelines, and second, an *Execution Engine* then is required to execute the respective series of Cognitive Applications and take care of the used and produced data in the context of the above semantic data infrastructure.

As has been mentioned before, the functional setup and interfaces of Cognitive Applications and the declarative, cognitive nature of their semantic descriptions (see again, Figure 7.5) are specifically designed to meet these requirements. Namely, the *composition* of miscellaneous algorithms into complex data processing pipelines only depends on the respective algorithms' *operations* and its *pre- and postconditions*, i.e., on their input/output annotations. The preconditions of a Cognitive Application state what data is needed for its potential execution, so one only needs to match them with the available patient data in the SFB knowledge base, and the postconditions state what data is produced by the respective algorithm, i.e., which data will potentially be uploaded to the SFB knowledge base after execution.

As all Cognitive Applications have been *registered* via Surgipedia in a common *Cognitive Application repository*, Surgipedia enables the *Pipeline Composer* to find and select suitable

Cognitive Applications (or rather their semantic descriptions) which *match* with the available data in XNAT and which hence fit for the further information processing in the context of a *data-driven pipeline*. For more information on the functionality of the *Pipeline Composer* and on its underlying Linked Data-based mechanisms to match data and algorithms, we refer to the works by Gemmeke [GMP⁺14] and Philipp [PMK⁺15].

Once – based on a cron job – a subsequent matching Cognitive Application has been found, it is the task of the **Execution Engine** (*Linked Data-Fu* [SSHS13]) to automatically *initialize* and *run* the respective algorithm, using appropriate parameters and the data and information stored in the knowledge base. Again, for more information on how the service application is initialized with adequate data and parameters, and how the algorithm then is executed, we refer to works by Gemmeke [GMP⁺14] and Philipp [PMK⁺15]. More details on *Linked Data-Fu*, which is the underlying framework for defining and executing rules to interact with semantically annotated Web resources, and which is hence used to create rules for data source integration, can be found in the work by Stadtmüller [SSHS13].

Lastly, the results created by a Cognitive Application (i.e., the actual result files *and* the attached RDF annotations which semantically describe these files) are directly uploaded to and saved in the semantic data storage again. As a consequence, newly uploaded files will be processed automatically in the next execution round of the *Pipeline Composer*. Also, the uploaded result files and their attached RDF annotations thus enrich the knowledge base by providing newly inferred, machine-interpretable information and enhanced patient data.

Summarizing, using a rule-based *Pipeline Composer* and *Execution Engine*, we enable the flexible composition and automated execution of complex pipelines of *Cognitive Applications* for medical information processing. With the thus constructed *declarative pipeline architecture*, we obtain loosely connected, cognition-guided, data-driven information processing pipelines and facilitate the integration and enhancement of new (semantic and non-semantic) data sources; see again, Figure 7.6.

How to create a Cognitive Application? - Semantic Wrapping, Implementation, Deployment and Registration. In order for a general medical information processing algorithm to become a *Cognitive Application*, and to then be integrated with the common *SFB knowledge base* so the above-mentioned *Pipeline Composer* can utilize it for the creation of complex medical information processing pipelines, this algorithm must go through a specific *semantification and deployment procedure*. In the following, this procedure will be presented in detail.

First, all information processing algorithms and services need to be *semantically wrapped* and *exposed as RESTful Web APIs*. Technical support for this task is given, e.g., by means of the Java API for RESTful Web Services (*JAX-RS*¹⁰) implementation *Jersey*¹¹. Implementation-wise, by using the *generic Cognitive Application implementation template* proposed by Philipp [PMK⁺15], in order for our MVR simulation-related algorithms, we are provided with a set of basic Java methods which facilitate – among others – the following essential tasks:

¹⁰ JAX-RS: *Java API for RESTful Web Services*, a Java programming language API spec that provides support in creating web services according to the REST architectural pattern, e.g., via the specific "@XXX" annotation pattern.

¹¹ Jersey: *Jersey RESTful Web Services framework*, an open-source framework for developing RESTful Web Services in Java. It provides support for *JAX-RS APIs* and serves as a JAX-RS reference implementation and central library.

(a) parsing incoming RDF descriptions, (b) querying XNAT for files (or, respectively, for their attached RDF annotations) needed for the service execution, (c) invocation of binaries and obtaining respective service results, (d) generation of new RDF annotation files for the resulting output of the algorithm, and (e) the upload of the semantically annotated service results to XNAT, as well as the generation of an RDF file that summarizes the service execution. In all these methods, the open-source framework *Apache Jena*¹² is used to support the processing of Linked Data and the execution of SPARQL patterns.

Principally, these generic methods only need to be adapted to the respective service that shall be wrapped and exposed as a Cognitive Application. This being said, the *semantic wrapping* of the algorithm entails that the service is extended by a semantic description according to the above-presented *algorithm and data ontologies*. It may be guided by means of the above-mentioned *Surgipedia algorithm annotation templates*, so the semantic markup hence comprises service properties like operation, functionality, internal mechanisms, classification, control parameters and input/output data types/formats. Using RDF notation, one can thus lift the service functionalities and its I/O interfaces to a semantic level.

Having gone through this procedure, the obtained Cognitive Applications then need to be deployed on a *Web Server*, e.g., using the open-source software *Apache Tomcat 6*¹³ which enables the emulation of a Web Server on an arbitrary workstation/computer. As such, compute-expensive simulation algorithms, which may require for the usage of HPC clusters and which hence need to be run on adequately efficient systems, can still provide their services via the Web when wrapped as Cognitive Applications, i.e., when being remotely accessible RESTful services.

Lastly, once being semantically wrapped as a RESTful service and deployed on a Web Server, the Cognitive Application needs to be *registered* via Surgipedia in order to become part of the common *Cognitive Application repository*. When having used the *Surgipedia algorithm annotation template*, this task has automatically been carried out as each Cognitive Application is allotted a Semantic MediaWiki page on Surgipedia. Alternatively, it is possible to write a pure RDF-based description of the algorithm, and upload and link it to an identically named Surgipedia page. Either of these is a prerequisite for the *Pipeline Composer* to be able to find and integrate the respective service into potential information processing pipelines.

Hence, Cognitive Applications are mainly wrappers (namely, semantically annotated wrappings) for medical interpretation algorithms, which are made available through RESTful Web interfaces. Their communication and interaction with the knowledge base and with the compute servers is based on Linked Data (Cognitive Applications are Linked Data consuming and producing Web applications), and – when being run on a Web Server – they may offer their functionalities/services to arbitrary, authorized clients within the assistance system.

For more details, we refer to the works by Gemmeke [GMP⁺14] and Philipp [PMK⁺15] who gave in-depth information and guidance on the overall framework, setup and usage of Cognitive Applications.

¹² Apache Jena: an open-source Semantic Web framework for Java, which provides an API to extract data from and write to RDF graphs, and to generally process Linked Data, e.g., using RDF parsers, or SPARQL.

¹³ Apache Tomcat 6: an open-source Web Server developed by the Apache Software Foundation (ASF), which implements several Java EE specifications and provides a pure Java HTTP Web Server environment in which Java code can run.

Summarizing, we have outlined the concept of a *semantic software architecture* that is based on *Cognitive Applications*. We have further shown how it is to integrate into the aforementioned setup of our common *semantic data infrastructure* and of its underlying knowledge base.

The proposed concept of Cognitive Applications, in combination with the previously introduced semantic data infrastructure, supports the integration of miscellaneous tools and algorithms for an automated processing and consolidation of heterogeneous medical data sources, and thus meets the above posed software requirements for a *cognitive surgery assistance system*.

Concluding, we emphasize that the opportunities for **Cognition-Guidance** are only enabled through the *overall combination* of Semantic Web and Cognitive Computing Technologies; namely, Linked Data, the Semantic MediaWiki (SMW) Surgipedia and the utilized ontologies, the Linked Data Crawler, the Triple Store and its SPARQL endpoint, the central XNAT patient data storage system, and medical information processing algorithms that are semantically wrapped as Cognitive Applications.

In the following Section 7.3, we will give evidence about the potential of the proposed overall concept of semantic data infrastructure and cognitive software components by the example of *simulation-enhanced MVR surgery assistance*. We will show that, as opposed to procedural pipelines, with the concept of Cognitive Applications, *declarative pipelines*, i.e., rule-based information processing pipelines, open up diverse options for Cognition-Guidance.

7.3 Sample Application: Cognitive Tools Pipeline for MVR Surgery Assistance

Having presented a concrete system architecture for Cognition-Guided Surgery Assistance in the previous Section 7.2, which comprises a common *semantic data infrastructure* and a concept how arbitrary software components can semantically be wrapped and thus interact with our knowledge base, now, we will continue with an exemplary cognitive application pipeline: our *cognitive, simulation-enhanced MVR surgery assistance setup*.

This setup seamlessly integrates into the joint cognitive surgery assistance system and exploits its knowledge base (as presented in 7.2.1), and it makes use of Cognitive Applications (as presented in 7.2.2) that semantically wrap dedicated software components for MVR surgery simulation. The respective single components and their underlying functionalities have been presented in the previous Chapters 4, 5 and 6.

As stated in Chapter 2, during MVR surgery, cardiac surgeons must account for large amounts of potentially MVR-relevant impact factors and preferably keep in mind all available relevant patient information (which not only consists of image data but also of all other sorts of heterogeneous text and parameter data) in order to make comprehensive decisions towards suitable surgery strategies.

However, surgeons often cannot consider all potentially available medical data due to simple time constraints and sometimes due to a lack of surgical knowledge or interpretation experience. Also, during traditional MVR, surgeons unfortunately often do not have the needed access to potentially valuable recommendations and data that might help for choosing the appropriate strategy. Drawing on a previous example, for a given MVR patient, it would be useful for the operating surgeon to know *how* other successful surgeries were performed on patients that had similar disease characterizations and *why* they were performed in that way. Hence, in such a

case, cardiac surgeons may clearly benefit from suitable appraisements or from information on recommended best practices from other accepted experts and colleagues in the field directly in the operating room.

New cognitive technologies as presented in Artificial Intelligence and Data Science research throughout the last couple of years (see again Section 7.1) and IT-based support in general thus seem promising, when it comes to the derivation of an *all-encompassing recommendation for a suitable surgical strategy*. This especially holds during complex operations, such as an MVR, where surgeons may be provided with useful additional (and hence, more comprehensive) information for improved diagnosis and therapy.

For this purpose, we aim at making use of the previously presented technologies and the cognitive system architecture in order for a *maximally holistic surgical information processing*, and, eventually, for an *intelligent, suitable MVR surgery simulation* as part of our prototypic simulation-based MVR surgery assistance.

To be precise, we have proposed in our work [SEZ⁺15] a concept for the integration of a biomechanical MVR surgery simulation into a knowledge-based surgery assistance system for MVR. Through an approach which uses surgical guidelines, that semantically implement MVR surgery strategies and expert experience, we perform reasoning on the available medical patient data in order to thus deduce a set of potentially suitable annuloplasty ring prostheses. Eventually, we set up patient-specific MVR soft tissue simulation scenarios which involve these rings, so surgeons can virtually investigate on their potential MVR surgery results *before* the actual operation.

In the following, we will specifically show how we can make use of the above presented *semantic system architecture*, which comprises both the *semantic data infrastructure* along with the introduced means for knowledge representation and, at the core, the central common *knowledge base*, as well as the *cognitive software components* which have access to and thus benefit from the represented knowledge and information. We will point out by the example of ***cognition-guided, simulation-enhanced MVR surgery assistance***, how we can exploit available patient data and surgical knowledge in order for a simulation-based enhancement of patient-individual cardiac surgery assistance.

Figure 7.8 illustrates the entire knowledge-based workflow architecture of the proposed cognition-guided, simulation-enhanced MVR surgery assistance system setup, which we proposed and specified in our works [SEZ⁺15, SKS⁺16].

It is built upon the common *knowledge base* [FMK⁺16] and hence continuously uses and feeds back data from and into it throughout the entire workflow. Using 4D-TEE-US images of the heart, we carry out semi-automatic *segmentations* of the entire valvular apparatus [ELAM⁺15], and further process these segmentations for subsequent simulation purposes, e.g., by means of annotation of important landmarks, or identification of specific quantities and properties. Besides that, non-imaging patient-individual MVR-relevant parameters and values are *digitalized* from clinical investigation reports and *semantified*, i.e., structured and converted into RDF [SPW⁺16]. They can then be analyzed and interpreted by a *deductive system*, which implements surgically motivated guidelines that are represented through an ontology. This enables the inference of potentially suitable ring prostheses for implantation during MVR [SPW⁺16].

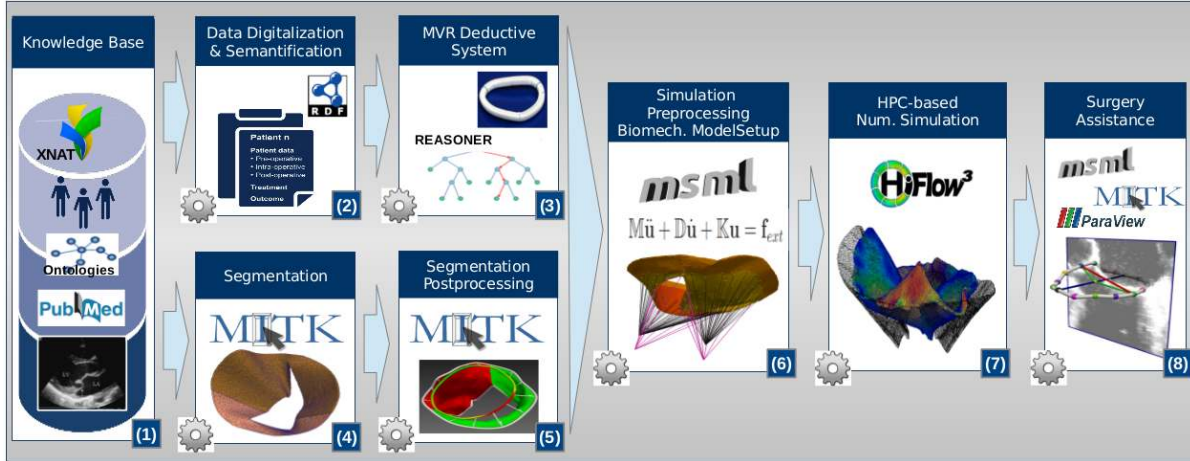


Figure 7.8: Workflow diagram and concept of the overall *cognition-guided, simulation-enhanced MVR surgery assistance system*. The algorithms and functionalities which underlie the respective building blocks have either been described in the previous chapters and/or were published and presented in journals, proceedings and on conferences each:

(1) The semantic data infrastructure and the setup of the common *Knowledge Base* was detailed in Subsection 7.2.1 and is described in our work [FMK⁺16].

Concerning non-image and parameter data, (2) the procedure for *Data Digitalization and Semantification*, as well as (3) the *Deductive System for MVR annuloplasty ring prosthesis selection* was presented in Schoch et al. [SPW⁺16]. With respect to the processing of image data, (4) the 4D TEE Ultrasound imaging-based semi-automatic *Segmentation Algorithm* and (5) the MITK-based *Segmentation Post-processing* step have been published by Engelhardt et al. [ELAM⁺15].

Merging the two information processing queues, (6) the *Simulation Preprocessing* component builds on top of the MSML (see Section 6.1 and our work [SSS⁺14]), and takes care of the proper biomechanical model and simulation scenario setup, as described in Section 6.2 and published in our work [SKS⁺16]. (7) The *Numerical Simulation Application* then executes the afore set-up MVR surgery simulation scenario as described in Chapters 4 and 5 as well as in our work [SES⁺15]. (8) A *Simulation Postprocessing and Surgery Assistance* step finally post-processes the simulation scenario results in a way that is beneficial to the operating surgeon, as presented in [SKS⁺15] and in Section 6.3.

The overall conceptual setup of this workflow setup for cognition-guided, simulation-enhanced MVR surgery assistance and the Cognitive Applications-based implementation and deployment (meaning that each block is wrapped as a Cognitive Application) was proposed and presented in our works [SEZ⁺15, SPW⁺16] as well as in Sections 7.2 and 7.3.

Resulting from these two information processing chains, a representation of the segmented MV geometry as well as the type and size of an annuloplasty ring which is suggested to be implanted during MVR is obtained. Based on this information, a biomechanical MVR soft tissue simulation shall accordingly be set-up in a *simulation preprocessing* step, defining a numerical simulation scenario which builds on top of a corresponding biomechanical MVR model [SKS⁺16]. The thus prepared patient-individual simulation scenario is then executed by our HiFlow³-based *biomechanical MVR simulation application* [SES⁺15], which virtually simulates the potential implantation procedure and the patient-specific post-operational behavior of the MV. Lastly, in the context of a simulation postprocessing step, important simulation results and quantifications need to be visualized and made available for the surgeon in the OR [SKS⁺15].

In the following, we will look at a subset of these building blocks in detail, and point out how they have been wrapped semantically, and how they thus integrate into the overall cognitive system architecture.

Data Digitalization and Semantification. With respect to non-imaging-related data, in a first step, patient-individual MVR-relevant parameters and values from clinical and medical investigation reports – that are recorded across different clinical departments and medical laboratories, commonly in a manual, hand-written way – need to be *digitalized*, i.e., transferred into some clinical reporting system or data base. This process may either entail text mining algorithms and reasonability checkers or parsers, or – and this is shockingly often still the case – be of purely manual nature, meaning that a human assistant transfers hand-written data into a data base or Excel sheet.

Important properties and MVR-relevant parameters include, for instance, valve and tissue characteristics (dilative, degenerative or rheumatic MV), size and shape of the annulus, curvature and elasticity of the leaflets, chordae rupture or integrity, or prolapse and mitral regurgitation (MR) properties (organic or functional MR).

After digitalization, the data is not necessarily machine-interpretable yet, such that a *semantification procedure* is required before any intelligent, machine-based further information processing can take place. In the case of our simulation-based MVR surgery assistance system, we make use of an *Excel2Rdf-Converter*, which converts the above-listed MVR-relevant parameters from a patient’s individual Excel sheet into a respective patient RDF file, where these parameters are structured and semantically represented according to the above-mentioned CGS ontology (the factor ontology, to be precise), see [SPW⁺16].

Resulting from this digitalization and semantification process (see block (2) in Figure 7.8), semantified patient RDFs are uploaded through the Cognitive Application interface to the XNAT medical patient data storage system (see block (1) in Figure 7.8), and hence available for further processing.

Deductive System for MVR Annuloplasty Ring Prosthesis Selection. Having semantified all MVR-relevant patient data, all information can be analyzed and interpreted by means of a machine now. We therefore download the data through our next Cognitive Application which wraps a *deductive system for MVR ring selection*; see block (3) in Figure 7.8. Similarly to a human surgeon in the OR, this deductive system analyzes the available patient data using

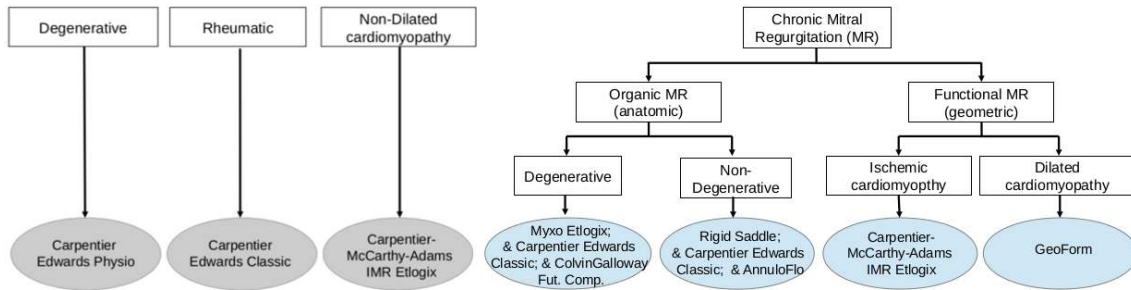


Figure 7.9: Decision tree representation of annuloplasty ring prosthesis selection guidelines according to the works of Carpentier [CAF10] (left/gray) and Fedak [FMB08] (right/blue): Depending on the patient’s conditions, i.e., the state of MVR-relevant patient parameters, a suitable (set of) annuloplasty ring(s) is selected for implantation during MVR.

established practical MVR surgery recommendations, and thus deduces an appropriate MVR surgery strategy along with a set of potentially suitable annuloplasty rings for implantation.

The underlying reasoning algorithm therefore interprets the given data according to so-called *surgical rules*, which implement MVR surgery guidelines from accepted MVR experts. These represent surgical knowledge and best practices in accordance with our common ontology. We exemplarily consider two highly esteemed publications on annuloplasty ring prosthesis selection; one from A. Carpentier [CAF10], the founder of modern reconstructive surgery, and the other one from P. Fedak [FMB08], which is also commonly cited in highly ranked journals.

From these works, we manually extracted *surgical rules* and translated as well as semantically represented them (using RDF with Notation 3 (N3) syntax) in a way that can be regarded similar to *decision trees*; see Figure 7.9.

As can be seen in Figure 7.9, we – so far – conduct reasoning on the basis of only few general MVR-relevant parameters only (see again, the list in the previous paragraph). However, we remark that the current set of parameters and the respective rules are flexibly extensible if further knowledge shall be considered. Also, it is obvious that diverse types of rules can be represented and implemented: for instance, iteratively nested rules, straight single-level rules or branching multi-level rules of arbitrary complexity – depending only on the level of sophistication, comprehensiveness and complexity of the considered MVR-relevant parameters.

Hence, building onto such a set of patient-individual MVR-relevant parameters (i.e., our set of *asserted facts*), we want to *infer* logical consequences by means of these *surgically motivated rules*. In our deductive system, we therefore employ the forward-chaining engine *cwm* (version 1.2.1) for querying, checking, transforming and filtering information. Eventually, in order to execute the deductive system, we specify the respective patient and the set of rules to be employed by means of cURL-based queries; see our joint work [SPW⁺16].

It is to be pointed out that through using a hierarchy of inference rules, we allow for starting the reasoning process from a comprehensive set of all commercially available ring prostheses, in order to then gradually *exclude* more and more *unsuitable rings* for the respectively given patient. One hence finally obtains a *maximally minimized set* of potentially suitable MVR ring prostheses for the given patient.

Please note that this setup of *hierarchical inference rules* and a *forward-chaining engine* makes the system very *robust* and *deterministic*; even incomplete patient parameter sets will enable to conduct reasoning, i.e., the exclusion of unsuitable rings. Moreover, its underlying decision tree structure is easy to understand and *transparent* for the operating surgeon, and inferred ring selection results can be reconstructed via the decision tree branch series. Finally, as mentioned above, based on the system’s flexible *extensibility* with respect to the consideration of additional MVR-relevant patient parameters and to the inclusion of further sets of surgical guidelines, the resulting propositions will likely become more and more reliable, too.

Knowing this, we want to emphasize that the proposed reasoning algorithm is to be considered as a *prototypic implementation* which does not claim for perfection in terms of performance, nor for completion or comprehensiveness in terms of the underlying parameters and inference rules. It rather is to be seen as general *proof of concept*, which is to hold as a basic reasoning step for application in the overall workflow for simulation-enhanced MVR surgery assistance. We clearly state that the presented system would need to be extended in many ways before any well-founded reasoning can take place. The most important way of extending would likely be the system’s capability to deduce the size of the respective ring type and the way of implanting it onto the natural annulus; see [SPW⁺16].

This being said, the result of the reasoning algorithm is a set of suitable annuloplasty ring prostheses, which – via the Cognitive Application interface – is uploaded along with an RDF annotation to the central XNAT file storage system, again, in order to allow access for the next tool in the pipeline.

Simulation Preprocessing. Knowing that a human surgeon’s decision neither only builds on top of image data, nor only builds on top of quantifications and parameter data, we will need to consolidate these two pieces of information, e.g., in order for an image-based optimal placement and implantation of the respectively specific ring. In order for a simulation to incorporate these patient-individual properties and the underlying surgical knowledge, the corresponding biomechanical model needs to be set-up appropriately.

For this purpose, we employ the *Medical Simulation Markup Language (MSML)* and the above-proposed *comprehensive MVR simulation preprocessing workflow*; see again Sections 6.1 and 6.2. We have wrapped and semantically described the entire MSML-based biomechanical MVR simulation preprocessing workflow as a Cognitive Application (represented through block (6) in Figure 7.8), and hence allow for the processing of both parameter and image data:

The Cognitive Application’s *precondition* is therefore twofold: On the one hand, we integrate patient-individual 4D TEE US-imaging-based annotated segmentations of the MV that were obtained in the two imaging-related preceding steps (see (4) and (5) in Figure 7.8) by means of the open-source imaging toolkit MITK. On the other hand, we include the information derived by the above depicted parameter digitalization and semantification step and by the deductive system for MVR ring selection (see (2) and (3) in Figure 7.8), namely, the set of suggested annuloplasty ring prostheses.

Thereon-based, we execute a series of *analytics tools and data manipulation operators* as part of a complex and comprehensive simulation preprocessing workflow, which was presented in detail in Section 6.2, and thus facilitate the *fully-automated* setup of *comprehensive, cognition-guided, patient-individual biomechanical MVR simulation scenarios*. For a visualization of a

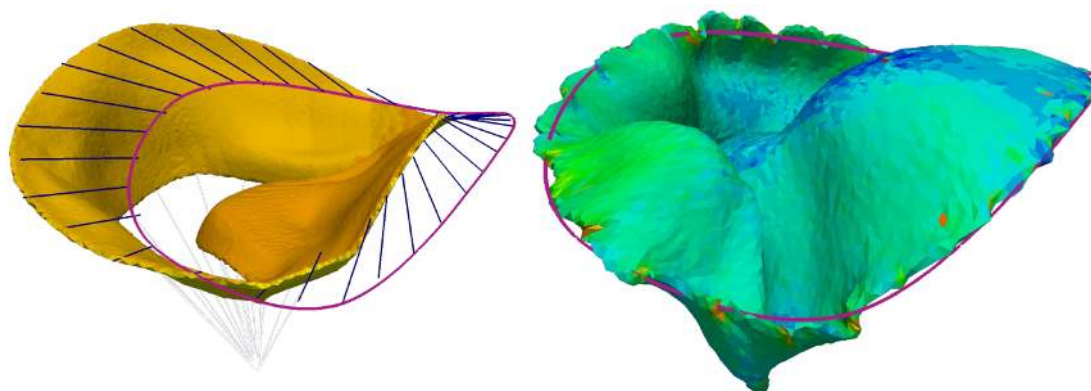


Figure 7.10: Results from two steps in the Cognitive Application Pipeline for Cardiac Surgery Assistance: (a) MVR soft tissue simulation scenario setup with a patient-individually chosen ring prosthesis, obtained from the MSML-based MVR Simulation Preprocessing block. (b) MVR soft tissue simulation scenario results showing the MV deformation and stress behavior after the virtual implantation of the chosen annuloplasty ring prosthesis, obtained by means of the MVR soft tissue simulation application based on HiFlow³, and post-processed through the MSML-based MVR Simulation Postprocessing block.

such obtained exemplary MVR simulation scenario setup, see Figure 7.10 (a). The geometric representations of the corresponding ring prostheses had been digitalized and optimally sized and placed with the help of MITK Mitralyzer before [GSAM⁺13, GWAM⁺14, EdSAM⁺16].

The resulting MVR simulation scenario setups – defined by the Cognitive Application’s *postconditions* – are suitable for subsequent simulation using our MVR soft tissue simulation application (as presented in Chapters 4 and 5). For every simulation scenario, we upload – via the Cognitive Application interface – a respectively resulting simulation scenario setup file, an MV mesh *inp* input file, a BCdata *xml* input file, and an RDF annotation to the central XNAT storage system again. These are thus made available for further processing and for execution by means of a subsequent Cognitive Application (see block (7) in Figure 7.8), whose *preconditions* are satisfied by the above *postconditions*.

MVR Surgery Simulation. The previously produced biomechanical MVR simulation scenarios subsequently need to be executed by means of our *HiFlow*³-based *MVR soft tissue simulation application* (see again, Chapters 4 and 5).

We have therefore also wrapped and semantically described the entire MVR simulation application as a Cognitive Application (see block (7) in Figure 7.8). The *precondition* is defined to consist of an MSML-set-up MVR simulation scenario, which is thus enabled to be remotely executed, e.g., on an HPC cluster.

The simulation scenario setup file is sent (along with the MV mesh *inp* and the BCdata *xml* input files) to an HPC cluster – in our case, the *bwUniCluster*¹⁴, which is located in Karlsruhe,

¹⁴*bwUniCluster*: see <http://www.bwhpc-c5.de/wiki/index.php/Category:BwUniCluster>

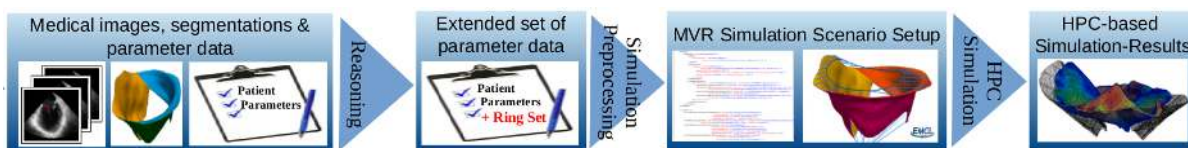


Figure 7.11: Evolution of medical data: Based on our Cognitive Application pipeline, we were able to gradually enhance and complete the medical information for an MVR test patient in our prototypic MVR surgery assistance system setup.

Germany –, on which the MVR soft tissue simulation application is installed and deployed along with the underlying FEM software toolkit HiFlow³.

Once the simulation execution is finished, the results are filtered and compressed in order to be transferred and – after semantic RDF annotation by means of the Cognitive Application wrapping – uploaded to the central XNAT file storage system. See Figure 7.10 (b) for an exemplary visualization of the simulation results for a respective MVR patient.

Simulation Postprocessing. The previously uploaded simulation scenario results on XNAT then need to be post-processed and prepared for visualization before a surgeon can really benefit from them. Therefore, we have wrapped our Simulation Postprocessing operators (which were presented in Section 6.3) as a Cognitive Application, too, and thus make their services available for remote execution; see block (8) in Figure 7.8.

Among others, firstly, we thus evaluate the von Mises stress behavior and project and visualize critical stress values on the MV surface. Secondly, we compute the size of the coaptation area in every visualized simulation time step. This enormously supports surgeons in the assessment of the potential quality of their intended MVR surgery strategy. Again, please see Figure 7.10 (b) for an exemplary illustration of post-processed simulation scenario results.

Being the last step in our Cognitive Application pipeline for Cardiac Surgery Assistance, we finally annotate and upload the post-processed simulation scenario results to the central XNAT patient data storage system, so they are available for usage in the OR during MVR surgery.

Concluding remarks. In order to fully automate the entire workflow for MVR surgery assistance, we now have all above-mentioned tools and algorithms integrated into the framework of a complex Cognitive Application pipeline; see again, Figure 7.8.

Therefore, making use of the *generic Cognitive Application implementation template* (as presented in Subsection 7.2.2), we semantically wrapped all algorithms as Cognitive Applications. Also, using RDF, we annotated their in- and output data (*pre- and post-conditions*) as well as their *functionalities* according to the *Surgipedia algorithm annotation template*. Before that, the respective algorithms had all been set-up in a way such that they were executable as command line tools and accessible via RESTful Web APIs. Moreover, we deployed the individual tools on suitable servers, and thus provided access to them as-a-service via the Web.

Being registered via Surgipedia – again according to the procedure described in Subsection 7.2.2 –, we were thus able to automatically compose all information processing algorithms into one complex *Cognitive Application pipeline for MVR surgery assistance*. This

pipeline starts – for a respective patient – from image and parameter acquisition, goes via an MVR reasoning system and via a simulation preprocessing operator sequence, which yields a patient-individual MVR simulation scenario, which in turn is then handed over to our dedicated MVR surgery simulation application, the results of which are finally post-processed and visualized in the OR; see again, Figure 7.8.

By executing the pipeline we can show that the whole setup and all of its individual steps are compatible with each other, and that the association of sets of inputs and outputs from different components works correctly (since the pipeline is automatically passed through without errors). Heterogeneous medical information – i.e., both imaging and parameter data – thus gets *consolidated and leveraged* for MVR surgery assistance. We illustrate the cognitive tools pipeline-based *evolution of medical data* with respect to some imaginary MVR patient in Figure 7.11.

7.4 Discussion and Outlook: Towards Semantic Simulation for Surgery Assistance

Summarizing, with our *cognitive tools pipeline for simulation-based MVR surgery assistance*, and with its component tools, we have contributed to the co-creation and shaping of the overall semantic architecture of our *cognitive surgery assistance system prototype*, which, in total, is a common effort of the collaborative research center *Cognition-Guided Surgery* (SFB TRR 125). It satisfies a comprehensive and representative set of clinical and medical requirements that are all motivated by and founded on top of real-world applications in clinical and surgical environments.

With our series of MVR-related tools and algorithms (all of which had been presented in previous chapters and sections), we have shown how arbitrary software components can be wrapped semantically, and how the resulting *Cognitive Applications* are thus enabled to intelligently process semantically annotated medical patient data and important surgical information fully automatically in the framework of a comprehensive surgery assistance pipeline. We refer back to Figure 7.8 for an illustration of the assistance workflow in the context of our sample MVR application.

Beyond that, we have indicated how the presented cognitive tools pipeline seamlessly integrates into the *overall joint cognitive surgery assistance system architecture*:

We demonstrated how it takes advantage of the great potential of the underlying *semantic data infrastructure* and how it exploits the common *knowledge base*, e.g, for modeling surgical expert knowledge or for representing and exchanging patient information.

Hence, by building a data-driven Cognitive Applications pipeline architecture on top of our semantic data infrastructure, and by thus connecting a set of highly-specific individual MVR surgery assistance-related software tools and algorithms, we have achieved a *prototypic surgery assistance setup* that allows to process and leverage patient data and surgical knowledge in order to infer new simulation-based knowledge for MVR surgery support; see again, Figure 7.11. Also, through the integration of all available information sources, we make a step towards a more comprehensive and holistic surgical therapy and treatment planning.

Evaluation and Validation. Concerning validation and evaluation, we refer to our joint work [FMK⁺16] for more details with respect to the semantic data infrastructure. Application-wise, several use-cases demonstrate the feasibility of the proposed approach: März et al. made use of the setup and developed a concept to enable the joint storage and processing of heterogeneous data from various information sources in order to facilitate the holistic surgical treatment strategy planning in *liver surgery* [MHW⁺15]. Similarly, our work on a cognitive, simulation-based MVR surgery assistance workflow successfully makes use of cognitive software components that seamlessly integrate with and exploit the semantic data infrastructure and its underlying knowledge base in order to support *cardiac surgery* [SPW⁺16].

The latter simultaneously holds for an evaluation of the cognitive software pipeline architecture. Furthermore, the original works of Gemmeke [GMP⁺14] and Philipp [PMK⁺15] each present an evaluation with respect to an application in the context of medical image preprocessing for brain tumor progression maps. Concerning the overall setup of our cognitive tools pipeline for simulation-enhanced MVR surgery assistance, we emphasize that an evaluation has so far only taken place in terms of correctness of the respective information processing applications and of the data-driven declarative workflow. Future work must include further investigation and optimization in this respect.

Summary and Outlook. In summary, we have shown – by means of our prototypic cognitive, simulation-based MVR surgery assistance system setup, and by means of other works that were reported on – the great potential of selected recent IT developments in the context of *comprehensive clinical information processing* and *simulation-based surgery assistance*. It is hence beyond dispute that IT-based support can provide surgeons with valuable additional information before, during and after an operation, e.g., through biomechanical surgery simulation scenario results.

Nevertheless, to date, biomechanical simulations are rarely established in surgery assistance or in the context of clinical research. Among others, on the one hand, this is certainly due to their *complexity*, which entails that handling the respective simulation environments and simulation tools is usually too difficult for non-simulation-experts, like surgeons and medical staff in the OR. Also, to best set up simulations and to maximally benefit from them, an in-depth simulation experience and numerical knowledge is needed. On the other hand, it probably is due to still unclear *agreements on model and simulation setups* (M&S), meaning that – with respect to the contained medical and physical terms and relations – neither the numerical simulation nor the underlying biomechanical model is commonly agreed upon among surgeons, engineers and M&S experts.

Up to this point, this work is dedicated to the specific use-case of MVR surgery, which we intend to support through providing surgeons with patient-specific biomechanical surgery simulation scenarios [SEZ⁺15]. We have set up a complex, integrated, automated information processing pipeline to process MVR-relevant patient data and surgical expert knowledge, and thus enable cognition-guided, patient-specific MVR surgery simulation [SPW⁺16].

For the conceptualization and setup of the respective information processing pipeline and the underlying MVR simulation application, we implemented *specific, dedicated solutions* and made use of our numerical expert knowledge and experience, e.g., in order to choose the best-suitable numerical solvers and preconditioners for the given problem formulation, or tune the

HPC setup for obtaining best computational performance.

Going *beyond MVR surgery*, of course, one can set up such surgery-specific simulation applications for different types of surgeries separately. However, using *general numerics knowledge* and a *structured representation of the experience with numerical algorithms and their properties*, it is possible to define strong, general dependencies between miscellaneous parameters in the context of the surgery-dependent mathematical model setup on the one side, and the derived simulation properties on the other side.

Accordingly, large parts of the MVR simulation application and of the gained experience concerning its setup and implementation could be transferred to other potential future surgery simulation setups: for instance, the biomechanical contact model and/or the analytical process of setting it up and calibrating it on the basis of morphological data and relations.

Towards Semantic Surgery Simulation. *Semantic technologies* seem very promising in this respect [MHC07]. Semantically describing analytical features, numerical algorithm properties and structural dependencies, and formulating explicit rules, e.g., in terms of decision trees – see, e.g., the setup illustrated in Figure 7.12 – can enable *automatically inferred simulation setups* for *arbitrary surgery simulation types*.

Also, knowing that some simulation properties and variables (or, respectively, their real-world counterparts) are subject to *measuring inaccuracies* and to general *uncertainties*, and keeping in mind that these can be modeled via probability distributions (see, e.g., the work of Schick et al. [SHLM14]), adding a semantic representation of probability distributions and uncertain parameters seems promising, too. In particular, this may be done with respect to the theoretical frameworks of *uncertainty quantification (UQ)*; see the boxes marked with the UQ attribute in Figure 7.12. Representing the respective stochastic distributions of these parameters and properties as part of the semantic model may easily allow to quantify the uncertainties in an emerging simulation result.

Hence, as part of an ongoing work [SSSVH16] and as a rather long-term vision, we proposed and started to – on the one hand – draw up a *structured collection* of quantified numerics expert knowledge and a *semantic description* of this knowledge, which comprises numerical methods, parameters, properties, and dependencies. On the other hand, we intend to make use of an ontological representation of anatomical and surgical terms, properties and relations, which shall eventually be linked to the before-mentioned numerics knowledge representation.

Miller et al. [SHM07] similarly suggested to represent modeling and simulation knowledge in the framework of an ontology. However, to the best of our knowledge, there is currently no approach published, which aims not only at semantically representing modeling and simulation knowledge by means of ontologies, but which also intends to link this representation with an underlying numerics properties decision tree, which in turn is linked itself via a set of defined correlations and rules to a corresponding semantic representation and decision tree structure of anatomical and surgical properties [SSSVH16].

With this setup and thereon-based semantic reasoning, the usability of and accessibility to surgery simulation toolkits may enormously be simplified, and general types of surgery simulations can be set-up fully automatically and with respect to terms and relations that are commonly agreed upon (in ontologies and rule frameworks) by surgeons and simulation experts.

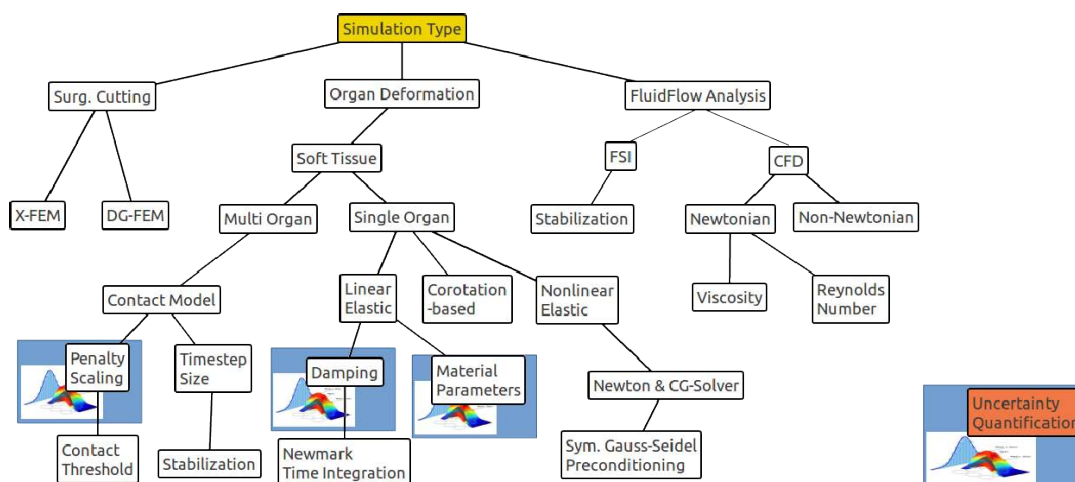


Figure 7.12: Sample dependencies tree to deduce simulation setup properties: In order to set up a simulation scenario, the tree is passed through from the root to the tip, and respective properties and parameters are accordingly set. Some parameters that can generally be subject to uncertainties are furnished with the UQ attribute, such that the respective stochastic parameter distribution (if known from experience and accordingly modeled) may automatically be included in order to thus be taken into consideration in the simulation.

According to the suggested methodology proposed in our work [SSSVH16], we have started to set up a basic *structured collection* and a simple *dependencies tree* representing numerics, modeling and simulation knowledge and respectively established connections or dependencies between miscellaneous parameters and properties. It basically extends the idea shown in the tree draft in Figure 7.12 and contains, among others, the following features:

- biomechanical models, e.g., elasticity, fluid flow, fluid structure interaction;
- boundary conditions, e.g., displacements, fixation, pressure, contact, flow profiles;
- model parameters and variables, e.g., damping, penalty scaling, material properties, sources for uncertainties and respective probability distributions;
- solution algorithms with respect to the FEM space discretization method, e.g., time integration methods (Crank-Nicolson, Newmark), solvers (CG, GMRES), preconditioners (Gauss-Seidel, AMG);
- methods for UQ, e.g., Monte Carlo or Polynomial Chaos extension;
- algorithmic properties, e.g., error behavior, convergence, stability, performance, accuracy;
- simulation-based optimization, e.g., description of goal functionals, min-max-dependencies, available/required HPC resources;

FMA:Liver	<i>is_a</i>	FMA:Organ		
	<i>consists_of</i>	FMA:SoftTissue	<i>has_property</i>	FMA:Stiffness
			<i>has_property</i>	FMA:Density
	<i>is_adjacent_to</i>	FMA:Lung	<i>is_part_of</i>	FMA:RespiratoryTract
	<i>has_image</i>	FMA:CTscan		
SoftBody	<i>has_a</i>	ElasticityModel	<i>is_a</i>	LinearElasticModel
			<i>has_a</i>	MaterialStiffness
	<i>has_a</i>	ContactModel	<i>has_a</i>	ContactBoundary
			<i>has_a</i>	PenaltyScaling
	<i>has_a</i>	Geometry	<i>originates_from</i>	CTscanSegmentation
Rule1:	Respiration	<i>causes</i>	PressureGradients	
Rule2:	PressureGradients	<i>can cause</i>	BoundaryConditions.	
Rule3:	SoftTissue	<i>can be</i>	Healthy or Pathologic.	
Rule4:	Healthy SoftTissue	<i>is less stiff than</i>	Pathologic SoftTissue.	

Figure 7.13: Sample extract of our prototypic semantic representations of medical or surgical entities/properties (upper box) and, respectively, numerics, modeling and simulation entities/features (middle box) on the implementation level. In addition to this, links and rules (bottom box) need to be defined according to these two semantic representations. Colors and arrows indicate these links and rules and hence illustrate how respective features and properties of the two ontologies couple with each other.

In describing the collected knowledge, we intend to primarily stick to the previously presented *modeling alphabet* of the *Medical Simulation Markup Language* (MSML) [SSS⁺14], and extend it with additional specific numerical features.

Yet, so far, we do not have a semantic representation of this interconnected numerics knowledge collection, which we proposed to be related and conforming with some upper ontology, namely, the Basic Formal Ontology (BFO) [ASS15]. Eventually, our numerics knowledge representation is to be derived and represented as a refinement of the BFO, as a *numerical modeling and simulation domain ontology* in order to thus allow for interconnection to and communication with other domain ontologies, such as a *surgical domain ontology*, namely, the Foundational Model of Anatomy (FMA) [RM07] which describes anatomically and surgically relevant properties.

Thereon-based, we proposed to set up basic *links and rules*, which constitute interconnecting dependencies between the *numerical modeling and simulation domain ontology* on the one hand, and the *surgical domain ontology* on the other hand. These links and rules need to be defined in accordance to the two respectively given ontologies, which, on the one hand, leads to commonly agreed upon modeling perspectives. On the other hand, they are to enable *semantic reasoning* via a semantic reasoning engine, which can execute the rules on patient data or respectively given queries in order to thus derive appropriate model and simulation scenario setups. See Figure 7.13 for an illustrating example, which has manually been implemented for a sample use-case in the context of liver surgery.

This setup hence enables numerical models and corresponding simulation scenarios to be inferred from anatomical or surgical situations, by simply going through the anatomical or surgical structured dependency tree, and through thus automatically having set the respectively

linked numerical property in the numerics properties decision tree, too. Properties or parameters which are not instantiated like this will be set to default values. Again, see Figure 7.13 for a liver surgery-related example on the implementation level.

With the completion of such a *semantic surgery simulation system* that is based on profound, structured numerics knowledge, one may expect to enable a general approach to set up patient-specific surgery simulations in the OR without any additional specific human numerics expert knowledge.

We hence aim at fostering and simplifying the usability of surgery simulations through our ongoing work and the implementation of this system, which comprises a *semantic representation of numerics and simulation properties* and an *underlying numerics properties decision tree*, which is *linked* to a corresponding *semantic representation of anatomical and surgical properties*.

The benefits of such an ontology-driven surgery simulation setup are likely to not only comprise a more time and cost efficient and largely automated simulation setup process, but also result from an expectedly increased simulation acceptance, which comes from the utilization of common modeling and simulation terms and relationships that were agreed upon in the framework of ontologies by surgeons and simulation experts [SHM07]. Through an ontologies-based setup, we also see a great potential of sharing and communicating our knowledge and experience, and of thus fostering the sustainability of the proposed approach. Finally, such a system enables improved objectification of simulation-based surgical treatment planning and patient care.

In the course of the implementation of a corresponding first properly operational prototype, we see the following *challenges* and *bottlenecks* to progress:

(1) Lots of *abstraction* work and respective *modularization* of miscellaneous simulation and simulation preprocessing software components is still needed. We therefore aim at learning from the concrete MVR case (see above) and at transferring and generalizing the therefrom gained experience.

(2) A *generalization* of the simulation properties in a research context is only achievable up to a certain level, as from which on further completion is a rather diligent but routine piece of work. We aim at achieving generalization for a few sample surgery scenarios to show general feasibility and flexibility.

For *evaluation*, one may, on the one hand, validate the complete setup by means of surgery application scenarios arising, e.g., in cardiac or liver surgery. On the other hand, the methodology and the single components can be evaluated separately, e.g., via reasoning and model comparison, i.e., through comparing the results of a numerics expert versus those from the reasoning system.

Finally, going even further, topping this ontology-driven surgery simulation setup system with *machine learning* (ML) capabilities seems very promising. Using established ML methods, one may conduct learning processes on the basis of patient or use-case similarities, and thus, eventually, improve the ontologically represented numerics knowledge and correspondingly inferred surgery simulation setups.

In this context, we also see the great potential of *Big Data* and *Big Data Analytics*. Large medical data sets from experiments, studies, and patient data bases, as well as a further growing list of medically and surgically relevant publications and experience-based guidelines compose an increasingly comprehensive collection of information and knowledge, which opens up new opportunities that wait to be exploited by means of adequate methods, e.g., in order to derive a respectively situation-dependent best surgical treatment strategy. Also, methods for *Data Assimilation* are considered auspicious when it comes to patient-individual calibration of simulation parameters and simulation scenarios; see again, paragraph **(C)** in Section 3.2.

For a good overview and further perspectives on Surgical Data Science, not only with respect to information processing for simulation-enhanced surgery assistance, we refer to the work of Vedula et al. [SVIH16].

8 Evaluation of the Overall Prototypic Simulation-enhanced Surgery Assistance Setup

In Chapters 4, 5, 6 and 7, we have presented the individual components and connecting links of the targeted prototypic simulation-enhanced cardiac surgery assistance system as well as its underlying semantic data and software architecture. Following, in this Chapter, we will look at the overall combined system setup and evaluate its functionality, efficacy and usability. Moreover, we investigate on the numerics and performance of its simulation component.

Firstly, in Section 8.1, we will present a *functional evaluation* of the overall simulation-enhanced MVR surgery assistance system prototype. The functional evaluation is based on the conceptual setup that was deployed in the context of the DFG assessment of the SFB TRR 125 project I03 *Functional Modeling and Numerical Simulation* in January 2016. According to the MVR surgery assistance setup presented in Section 7.3, in this Section, we will go through the entire information processing cognitive tools pipeline for one test patient. The presented results will indicate the achievement and efficacy of the overall design of our simulation-enhanced cardiac surgery assistance prototype, and give proof of the successful integration of a biomechanical FEM-based MVR surgery simulation into a cognition-guided surgery assistance system.

Secondly, in Section 8.2, we will *numerically analyze* the biomechanical MV/MVR model and our thereon-based FEM simulation application, and look at its HPC-optimized, multi-interface implementation in order to then conduct a *performance evaluation* with respect to simulation performance, efficiency, patient-specificity, accuracy and reliability. We therefore go through important simulation features and specifications, and present our experimental performance studies on an HPC cluster.

Concluding, in Section 8.3, we then suggest a short outlook with respect to further evaluation and validation opportunities for the presented setup.

8.1 Functional Evaluation of the Overall Surgery Assistance Setup

In this section, we evaluate the prototypic simulation-enhanced MVR surgery assistance setup presented in Section 7.3 in terms of its *conceptualization* and *functionality*. We therefore go through the entire medical information processing pipeline as shown in Figure 7.8, and indicate how medical patient data and surgical knowledge is automatically processed in order to set up comprehensive MVR surgery simulation scenarios that are executed and postprocessed for surgery assistance.

In going through the pipeline, we present the results of all components and hence of the overall pipeline by the example of *one test patient* and its medical data. Please note that we have artificially produced a *total of 49 test cases for the validation* of the *entire* surgery

assistance pipeline, however, will only exemplarily cover one of these in the following for going through the pipeline.

The total of 49 test cases for the entire pipeline was derived as follows: With the help of our clinic partners in Department of Cardiac Surgery at the University Hospital in Heidelberg, ultrasound imaging-based segmentations of MVs were obtained for two patients with severity I-III of MV insufficiency; see [ELAM⁺15]. For one of these two patients, they recorded 27 frames per cardiac cycle, and for the other one 10 frames per cardiac cycle. The average dimensions of the obtained DICOM images were $233.6 \times 217.6 \times 214.4$ voxels, with $0.81 \times 0.80 \times 0.74$ mm³ voxel size. For the first patient, out of 27 frames 13 were marked ‘suitable’ with respect to a subsequently imposed input restriction which requires that the segmentation provides an MV geometry with two properly separated leaflets that do not touch each other; for the second patient, 6 frames were marked ‘suitable’. Further, for the thus obtained set of $13 + 6 = 19$ MV segmentations from [ELAM⁺15], the respectively given sets of patient parameters (which were also recorded in the Department of Cardiac Surgery at the University Hospital in Heidelberg, and which were then transferred into the SFB knowledge base) indicated a potentially successful and reasonable implantation of three different annuloplasty ring prosthesis types: the *Carpentier Edwards Physio II* type (available for the sizes 32, 36, 40), the *Edwards IMR ETlogix* type (available for the sizes 32, 36), and the *Edwards Myxo ETlogix* (available for the sizes 28, 32). These rings were derived according to MVR surgery guidelines by A. Carpentier [?] and P. Fedak [FMB08] (as described above and in our work [SPW⁺16]), and geometrically represented for subsequent integration into 3D models by [GWAM⁺14]. Thereon-based, we were able to further process the available information through the entire MVR surgery assistance pipeline, and consequently obtained $19 \times 7 = 133$ MVR surgery simulation scenarios with respectively different ring prostheses out of the MSML-based simulation preprocessing step. Of these 133 potential simulation scenarios, only $133 - (12 \times 7) = 133 - 84 = 49$ were executed by means of our MVR soft tissue simulation application, whereas the other $(12 \times 7) = 84$ were directly aborted manually for scenario redundancy reasons (as these would obviously produce similar results for the respective patients).

Ultrasound Imaging and Segmentation. For the chosen test patient, we start from, respectively, 4D TEE Ultrasound imaging-based segmentations of the MV, and a set of patient-individual MVR-relevant patient parameters. Both were acquired and recorded in the framework of the SFB TRR 125 by surgeons and staff of the Department of Cardiac Surgery at the University Hospital of Heidelberg [ELAM⁺15, SPW⁺16].

Figure 8.1 shows a screenshot of the 3D visualization of the first time step (out of 27) of a set of *4D TEE US images* of the respective patient’s left ventricle and the MV. Using the medical imaging software MITK [NZS⁺13] and a semi-automatic *Segmentation* algorithm [ELAM⁺15], segmentations of the MV throughout the complete cardiac cycle were obtained. These segmentations were post-processed and annotated with landmarks and IDs, see Figure 8.2, such that, on the one hand, the two MV leaflets can be distinguished by means of two different IDs according to [ELAM⁺15], and in order to, on the other hand, identify important landmarks on the annulus and ring according to [GWAM⁺14]. Following this procedure, 16 control points describe the shape of the natural annulus and of the artificial ring in a way which corresponds to our MV prosthesis model as presented in Section 6.2. See again Fig-

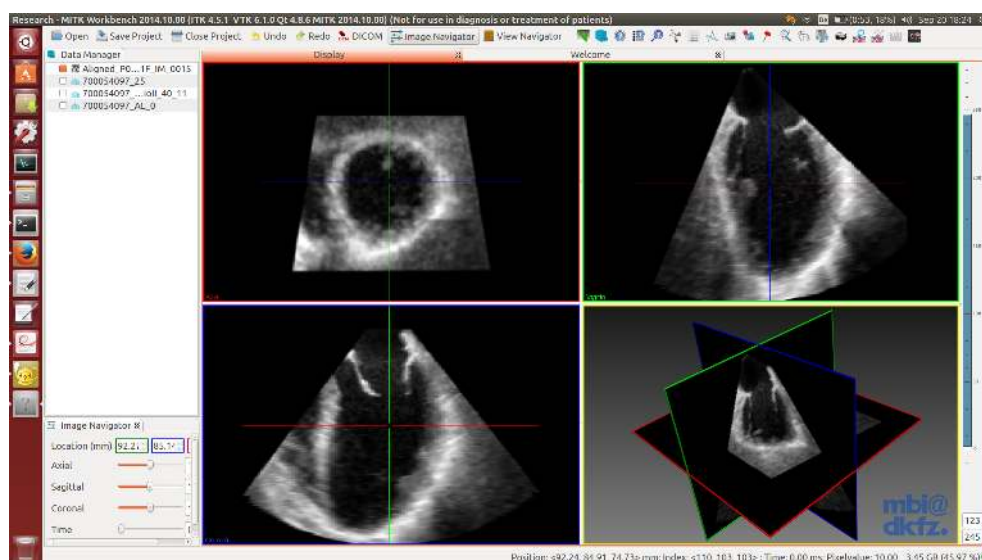


Figure 8.1: Screenshot of the 3D visualization of our test patient’s 4D TEE Ultrasound images using the software MITK. One can clearly recognize the patient’s left ventricle and the MV in the open state (time step 1 out of 27 which cover the complete cardiac cycle).

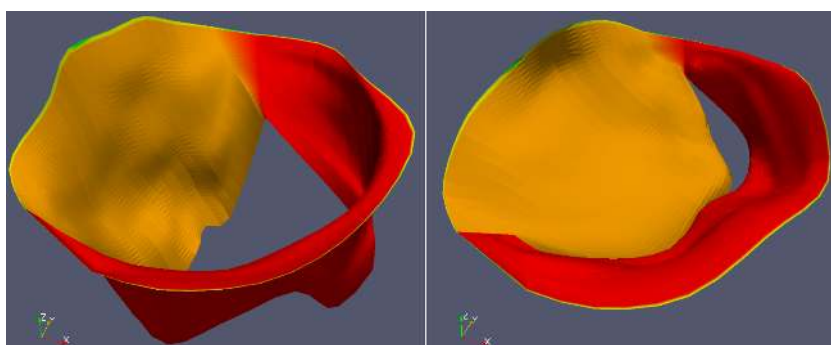


Figure 8.2: From the 4D TEE ultrasound images shown in Figure 8.1, segmentations of the MV were obtained for 27 steps throughout the complete cardiac cycle. The screenshots show two out of these 27 steps; the left one shows the time step when the MV is maximally open, the right one when it is maximally closed. Postprocessing the segmentations as described in the text provides the (color-coded) separation of the two leaflets and the identification of 16 important control points along the annulus (and the artificial ring prosthesis, respectively).

ure 6.4 for a visualization of the MV and MV prosthesis model, and Table 6.1 for the associated ID keys. Lastly, semantic annotations for both the 4D TEE ultrasound image of the patient’s MV and the MV segmentation are produced, and the respective data set is transferred into the central semantic XNAT file storage system [FMK⁺16].

Category 3	CG Factor (englisch)	Possible observation values	Observation	Unit (normal Range)*
mitral valve commissures lesions	mitral commissural fusion	yes, no	no	# boolean
mitral valve commissures lesions	mitral commissural thickening	yes, no	no	# boolean
mitral valve commissures lesions	mitral commissural calcification	yes, no	no	# boolean
mitral valve annulus morphology	mitral annulus shape	symmetric, asymmetric, ovoid, round	# one of list	
mitral valve annulus lesions	normal mitral annulus	yes, no	yes	# boolean
mitral valve annulus lesions	mitral annulus calcification	yes, no	no	# boolean
mitral valve annulus lesions	mitral annulus dilatation	yes, no	no	# boolean
mitral valve annulus lesions	mitral annulus sclerosis	yes, no	no	# boolean
mitral valve annulus lesions	mitral annulus infection	yes, no	no	# boolean
mitral valve annulus lesions	mitral annulus abscess	yes, no	no	# boolean
chordae tendinae lesions	mitral chordae rupture	yes, no	no	# string list
chordae tendinae lesions	mitral chordae calcification	yes, no, A1, A2, A3, P1, P2, P3	no	# string list
chordae tendinae lesions	mitral chordae sclerosis	yes, no, A1, A2, A3, P1, P2, P3	no	# string list
chordae tendinae lesions	mitral chordae shortening	yes, no	no	# string list
chordae tendinae lesions	mitral chordae elongation	yes, no	yes	# string list
chordae tendinae lesions	mitral chordae thickening	yes, no, A1, A2, A3, P1, P2, P3	no	# string list
chordae tendinae lesions	mitral chordae fusion	yes, no, A1, A2, A3, P1, P2, P3	no	# string list
papillary muscle morphology	normal mitral papillary muscles	yes, no	yes	# boolean
papillary muscle morphology	mitral papillary muscle calcification	yes, no, anterolateral, posteromedial	no	# boolean
papillary muscle morphology	mitral papillary muscle elongation	yes, no, anterolateral, posteromedial	no	# boolean
papillary muscle morphology	mitral papillary muscle rupture	yes, no, anterolateral, posteromedial	no	# boolean
ventricular lesions	ventricular aneurysm	yes, no, anterolateral, posteromedial	no	# boolean
ventricular lesions	ventricular fibrouse plaque	yes, no	no	# boolean
ventricular lesions	ventricular dilatation	yes, no	no	# boolean
ventricular lesions	myocardial tumors	yes, no	no	# boolean
ventricular lesions	infarction	yes, no	no	# boolean
ventricular lesions	myocarditis	yes, no	no	# boolean
ventricular lesions	ventricular calcification	yes, no	no	# boolean
special findings	special findings	yes, no	no	# boolean
special findings	special findings specification		# free text	
leaflet prolapse	mitral leaflet segment prolapse	no, A1, A2, A3, P1, P2,P3, AC,PC	# many of list	
annulus morphology	mitral annulus commissural diameter		# mm	
annulus morphology	mitral annulus septolateral diameter		# mm	
chordae tendinae morphology	anterolateral chordae length		# mm	
chordae tendinae morphology	posterolateral chordae length		# mm	
redo cardiac surgery	redo cardiac surgery	yes, no	no	# boolean
times	start of surgery		10:04	# time

Figure 8.3: Excerpt from our test patient’s set of MVR-relevant parameters; listed and represented in an Excel spreadsheet.

MVR Parameter Digitalization and Semantification. On the non-imaging data side, Figures 8.3 and 8.4 show excerpts from the patient’s individual set of MVR-relevant patient parameters. Figure 8.3 shows a screenshot from an Excel spreadsheet, in which surgeons and staff in the clinic record and digitally represent important data in the context of an MVR therapy. In order to further process this data and information by means of a machine (and, in particular, through a semantic reasoning system), the data in the spreadsheet gets semantically represented and annotated according to the common CGS ontology (and, in particular, the parameter and guideline ontology, see Figure 7.7). Resulting from the *Digitalization and Semantification* step, one then obtains an RDF file that contains the respectively patient-individual MVR-relevant parameters; see Figure 8.4 for an excerpt. Again, all semantically annotated data files are stored in the common semantic XNAT file storage system and hence in the SFB knowledge base; see again Section 7.2.1 and [SPW⁺16].

MVR Deductive System. As presented in Section 7.3, next, the patient’s parameter data is processed through the *MVR Deductive System* step for annuloplasty ring prosthesis selection. Resulting from the involved reasoning algorithms, one obtains a list of potentially suitable annuloplasty ring prosthesis types, that is suggested to the surgeon, and further processed via the MSML in order to allow for ring-specific simulation scenarios. For our test patient, the deductive system for annuloplasty ring selection suggests the implantation of the *Carpentier Edwards Physio II* annuloplasty ring prosthesis; see Figure 8.5 which shows the correspondingly extended patient RDF file. For the respective ring type, 3D geometric models (i.e., 3D representations in VTP format) in various sizes (here: 32, 34, 36, 40) are stored in the central XNAT storage system and get accordingly linked in the patient RDF.

```

PID0067_MVR-Param_E...2RDF-output_Nico.ttl ×
hft-version:operation%20time rdfs:label "operation time" ;
  a skos:Concept , sp:Category:Heart_Factor .

_:node19cr1b3hux53 sp:Property:possibleObsValue "number" ;
  sp:Property:has_Datatype hft-version:int ;
  sp:Property:has_Patient "ABCDE1234" ;
  sp:Property:has_Heart_Factor_Table hft-version:1.0.0 .

_:node19cr1b3hux54 a sp:Category:Heart_Factor_Observation ;
  sp:Property:has_Heart_Factor hft-version:endocardial%20thickening .

hft-version:endocardial%20thickening rdfs:label "endocardial thickening" ;
  a skos:Concept , sp:Category:Heart_Factor .

_:node19cr1b3hux54 sp:Property:possibleObsValue "yes" , "no" ;
  sp:Property:has_Datatype hft-version:one%20of%20list ;
  sp:Property:has_Patient "ABCDE1234" ;
  sp:Property:has_Heart_Factor_Table hft-version:1.0.0 .

_:node19cr1b3hux55 a sp:Category:Heart_Factor_Observation ;
  sp:Property:has_Heart_Factor hft-version:thrombus%20formation .

hft-version:thrombus%20formation rdfs:label "thrombus formation" ;
  a skos:Concept , sp:Category:Heart_Factor .

_:node19cr1b3hux55 sp:Property:possibleObsValue "yes" , "no" ;
  sp:Property:has_Datatype hft-version:one%20of%20list ;

```

Figure 8.4: Excerpt from our test patient’s set of MVR-relevant parameters; represented in RDF (Turtle) format.

```

PID0067_MVR-Param_D...tSys-Output_Nico.ttl ×
[...]
[
  a sp:Category-3AHeart_Factor_Observation;
  sp:Property-3Ahas_Heart_Factor hft-version:eligible_ring;
  sp:Property-3Ahas_Patient "ABCDE1234";
  sp:Property-3AobsValue "Carpentier-Edwards Physio II annuloplasty ring prosthesis" ].

```

Figure 8.5: Excerpt from our test patient’s set of MVR-relevant parameters, now extended by means of the deduced information on a potentially suitable annuloplasty ring prosthesis; represented in RDF (Turtle) format.

MVR Simulation Preprocessing. Using the thus obtained information and combining it with the patient’s annotated MV segmentation, next, our *data-driven MVR surgery assistance pipeline* automatically executes the MSML-based comprehensive MVR Simulation Preprocessing Workflow [SSS⁺14, SKS⁺16]. Resulting from this step, one fully automatically obtains the setup of a numerical simulation scenario of an MVR surgery for the given test patient and under consideration of the selected annuloplasty ring prosthesis. To be precise, as detailed in Section 6.2, the output of this step consists of three original files – (1) the HiFlow³ simulation scenario description xml input file, (2) an interlinked MV mesh geometry inp input file, and (3) a BCdata xml input file. See Figure 8.6 for an illustration of the thus obtained biomechanical MVR model, which allows the surgeon to easily understand the modeling features that govern the subsequent simulation results, which in turn enormously fosters the transparency and acceptance of our model and simulation. In addition to these original, MSML-produced files, the corresponding Cognitive Application creates an appropriate RDF annotation file. All of them are finally uploaded to the central semantic XNAT file storage system.

The entire simulation-enhanced MVR surgery assistance pipeline, which the MSML-based

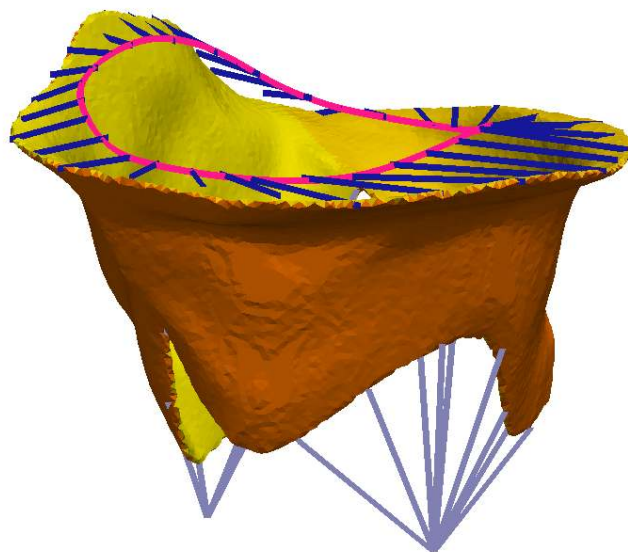


Figure 8.6: Visualization of the output of the MSML-based MVR simulation preprocessing workflow: an MVR soft tissue simulation scenario for the given patient under consideration of the selected annuloplasty ring prosthesis.

comprehensive MVR Simulation Preprocessing Workflow is part of, is not yet set-up and employed in a clinical testing environment. Hence, clinical validation has not yet taken place – neither of the entire system, nor of its partial components. Consequently, the efficacy of the simulation preprocessing pipeline or of the overall system cannot be proved.

However, the simulation preprocessing pipeline was designed as part of the MSML framework, such that, provided suitable input from the above presented steps in our pipeline, i.e., namely from the deductive system step and from the segmentation step, we can guarantee its *internal compatibility* and *error-free, fully-automated execution*, e.g., by type checking, automated conversions where needed, and an operator-based linking of input and output data. For more information, please see again, Section 6.2.

Besides the presented exemplary test patient, for validation, the MVR simulation preprocessing pipeline was run for 76 test scenarios in total, and executed as an integrated MSML-based workflow, using MV segmentations from the above segmentation step and respectively selected annuloplasty ring prostheses as selected from the above deductive system. Through the thus comprehensive set of test scenarios we are convinced to likely cover the range of possible input/output situations.

The resulting MVR Simulation Preprocessing output data always was fully compatible with our HiFlow³-based MVR Simulation Application input interface, and could directly be executed; see the below paragraph. This indicates the *robustness* and *efficacy* of our MVR Simulation Preprocessing software components.

Looking at the *performance*, empirical evaluation showed that the average time for the (sequential) execution of the complete MVR Simulation Preprocessing step amounts to roughly 50 to 80 seconds even on a laptop computer, mainly depending on the level of refinement of the

FEM volume mesh. Knowing that the MVR Simulation Preprocessing step is part of our overall *data-driven* MVR surgery assistance pipeline (meaning that a respectively next pipeline step is executed as soon as the required input data is available from the preceding pipeline step), one obviously finds that the preprocessing performance is not critical as the entire step is *performed preoperatively*.

Access to the open-source implementation of our MVR Simulation Preprocessing pipeline and to two sample test cases along with their respective preprocessing results is provided on: [GitHub.com/CognitionGuidedSurgery/msml/tree/experimental/](https://github.com/CognitionGuidedSurgery/msml/tree/experimental/).

MVR Soft Tissue Simulation. Based on the data produced by the above MVR Simulation Preprocessing step, i.e., a patient-individual HiFlow³-compatible simulation scenario description `xml` input file, an MV mesh geometry `inp` input file, and a BCdata `xml` input file, next, the *HiFlow³-based MVR Simulation Application* is executed to run the respective MVR surgery simulation scenario [SES⁺15]. For a detailed numerical evaluation of the simulation, we refer to the following Section 8.2.

As motivated in Section 4.2.1.2, the setup of our FEM-based MVR surgery assistance simulation is subdivided into two parts: part 1 performs the virtual implantation of the annuloplasty ring, and part 2 then simulates the post-surgery MV behavior for the closing part of the cardiac cycle. Resulting from the completed simulation step, one hence obtains patient-individual simulation results that show the virtual annuloplasty ring prosthesis implantation process, *and* the behavior of the MV during the cardiac cycle *after* the virtual implantation of the respectively selected annuloplasty ring prosthesis.

Using our HiFlow³-based MVR Simulation Application, the post-MVR MV deformation and stress behavior was successfully simulated. Figure 8.7 shows the simulation results for our test patient. The upper row depicts the natural morphology of the test patient’s MV, and the way in which the artificial ring prosthesis (in violet) is to be implanted by means of a suture (black lines) onto the natural annulus. The second row then visualizes the simulated implantation process of the patient-specifically selected annuloplasty ring prosthesis. The two lower lines (from left to right and then top to bottom) represent the MV closing behavior *after* the virtual annuloplasty ring implantation, i.e., the systolic part of a post-operative cardiac cycle. The simulation results show a restored closing functionality (*coaptation*) of the MV after corresponding MVR surgery and thus indicate the promising potential of such a minimally-invasive operation. Moreover, the von Mises stress distribution is color-coded and projected onto the leaflet surface in order for a surgeon to better assess the potential stress-caused risk of, e.g., a reoperation.

In our simulation and the underlying biomechanical model, all tissues were modeled as *Saint-Venant-Kirchhoff materials*, i.e., assumed to be homogeneous, isotropic, objective, and linear-elastic; compare Section 4.2. The respective *material parameters* are chosen according to classified average ranges. For our test patient, empirically evaluated average tissue properties as given by Mansi et al. [MVG⁺12] for patients suffering from *ischemic/functional mitral insufficiency* were employed. These include the *tissue density* $\rho = 1.04 \frac{\text{g}}{\text{mL}}$, and the two Lamé constants $\lambda = 56\,933 \text{ Pa}$ and $\mu = 1\,400 \text{ Pa}$. *Damping* was applied according to the approach of *Rayleigh*, see again Section 4.1.3, with $\alpha = 0.3s^{-1}$ and $\beta = 0.3s$, thus increasing the overall system robustness especially under strong collisions when the leaflets get into contact.

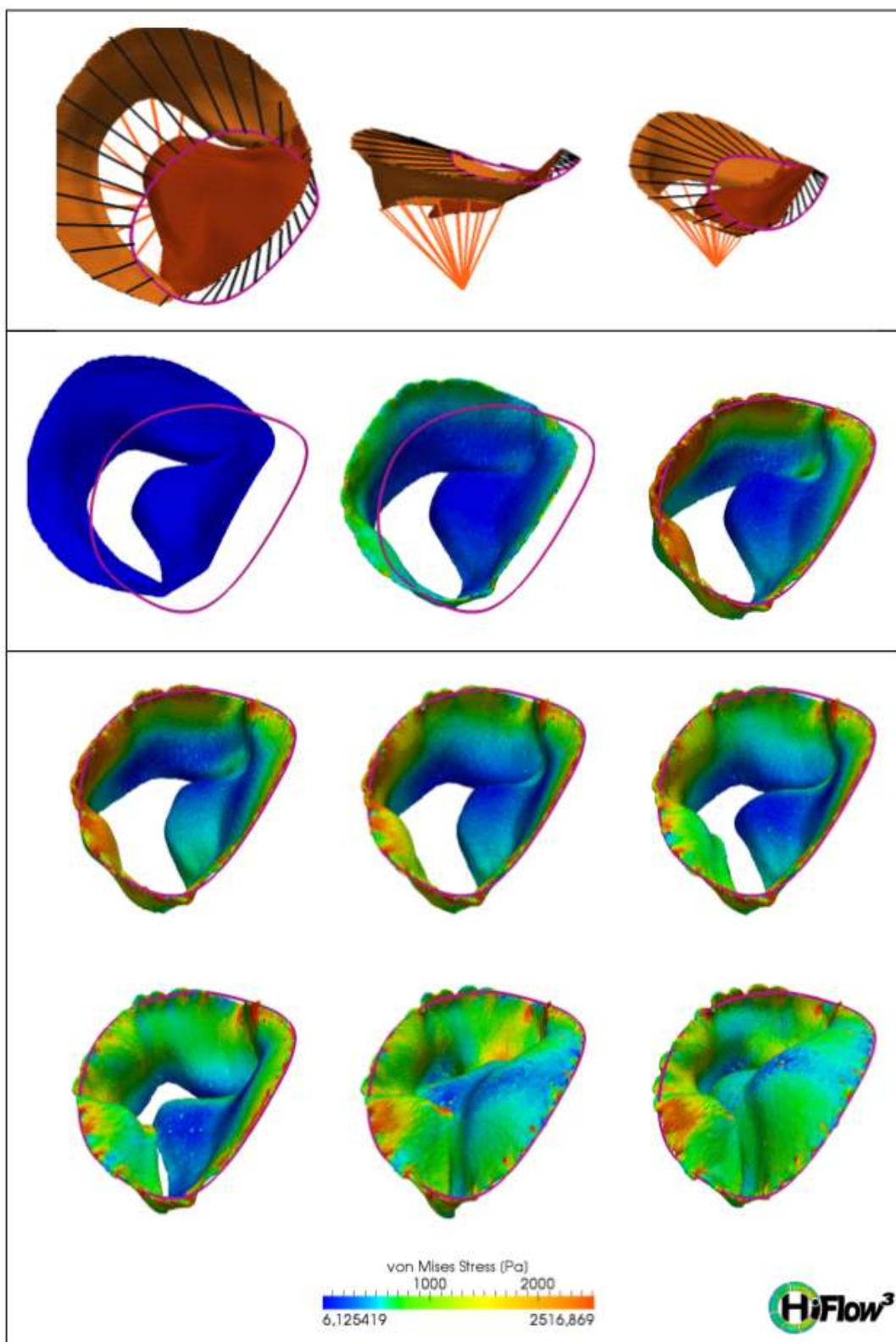


Figure 8.7: MVR simulation results: Visualization of the virtual annuloplasty ring implantation process and the post-surgery MV behavior as described in the text.

Apart from the material properties, we fully automatically obtained the setup of the *boundary conditions* (BCs) from our comprehensive MVR simulation preprocessing workflow step; see the preceding paragraph.

On the one hand, we have BCs for representation of the annuloplasty ring prosthesis implantation process: According to our prosthesis model depicted in Figure 6.4, we have the natural annulus virtually sutured to the artificial ring prosthesis by means of *pointwise Dirichlet displacement BCs*. See again Section 4.2.1.2 for more details. Being part of the first step in our simulation, the virtual ring implantation is scaled to take place in a 5 seconds time frame.

On the other hand, we have BCs to describe the forces and momentums arising within the entire ventricular apparatus during a cardiac cycle after MVR surgery: These include time-dependent *surfacewise Neumann BCs* on the MV leaflets to reproduce the effects of the blood pressure throughout a cardiac cycle. A physiological transvalvular pressure drop, up to 120 mmHg, is applied on the ventricular side of the leaflets. In addition, there are *pointwise Neumann BCs* to imitate the elastic effects of the (primary) chordae tendineae, and *surfacewise Contact BCs* to account for the event of leaflet contact. For more details, we refer to Section 4.2.1.1. As suggested by Votta et al. [VCV⁺08], the respective simulation time for the post-MVR cardiac cycle is artificially prolonged to 10 seconds (as opposed to 70 ms to 150 ms in reality) in order to handle the strong and discontinuous Neumann and Contact forces.

With the respective simulation results we expect to be enabled to strongly support the surgeon in the ring selection step during MVR surgery, as, for instance, *suboptimal rings* may be *detected* through virtual simulation *before* the actual operation. To demonstrate and evaluate the effect of the proper ring selection on the restoration of leaflet coaptation, the virtual implantation of a slightly smaller annuloplasty ring (with size 32 instead of 36) is depicted for our test patient in Figure 8.8.

As can be seen, the smaller ring unveils a totally different leaflet closing behavior. At certain parts of the annulus, the tissue is heavily serrated along the artificial ring (marked with the red circle in the upper part of Figure 8.8), such that an effect similar to the behavior of corrugated cardboard occurs, which enormously stiffens the leaflet tissue, and prevents it from properly making contact and from satisfactory coaptation (also marked with a red circle). The leaking coaptation is also revealed in the lower part of Figure 8.8, which shows two parallel slices cut vertically through the MV in order for an easier assessment of the coaptation zone. Again, a red circle highlights the critical behavior where a lack of contact can be observed. Corresponding results with respect to the potential stiffening of the MV leaflets in the neighborhood of sutures which strongly serrate the tissue after MVR have also been reported by Ge et al. [GMW⁺14] and by Stevanella et al. [SMC⁺11].

Please keep in mind, that the presented simulation results, however, need to be treated with caution: both the patient data and the applied simulation model are subject to uncertainties and partly incomplete. For instance, as has been mentioned above, the chordae tendineae and the papillary muscles both have a significant impact on the leaflet behavior, however, their behavior during the cardiac cycle is not yet accessible through the imaging and segmentation techniques of our clinical partners. Also, the underlying model which is based on linear elasticity is not completely suitable and should be replaced by some more advanced formulation in the future. Therefore, the simulated behavior may partly deviate from the real behavior, both with respect to the deformation behavior and to the stress development.

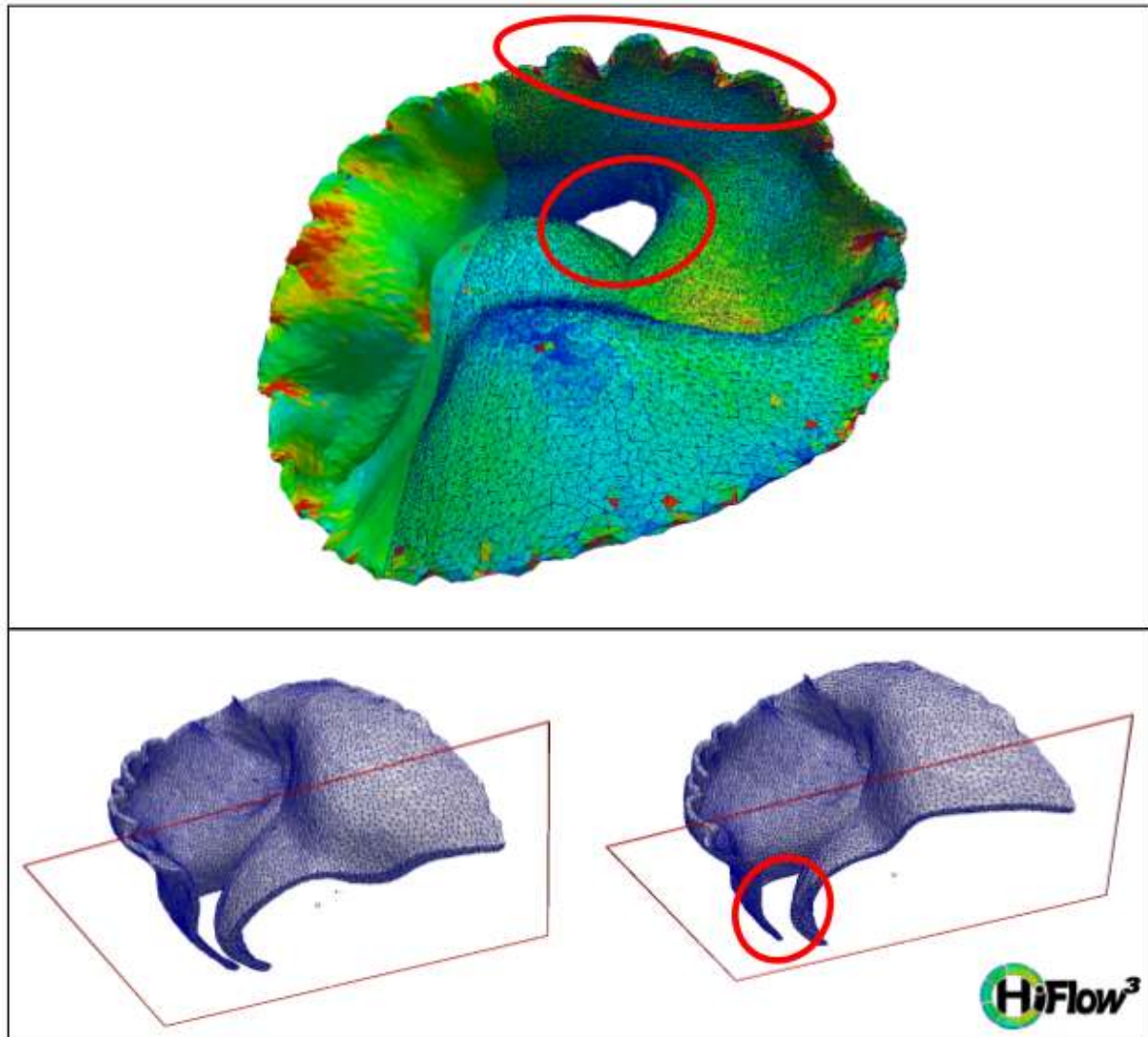


Figure 8.8: MVR simulation results: Visualization of a suboptimal, potentially critical annuloplasty ring implantation scenario, where a too small ring serrates the tissue too much (*upper part*), such that it becomes too stiff and does not enable proper coaptation anymore (*lower part*).

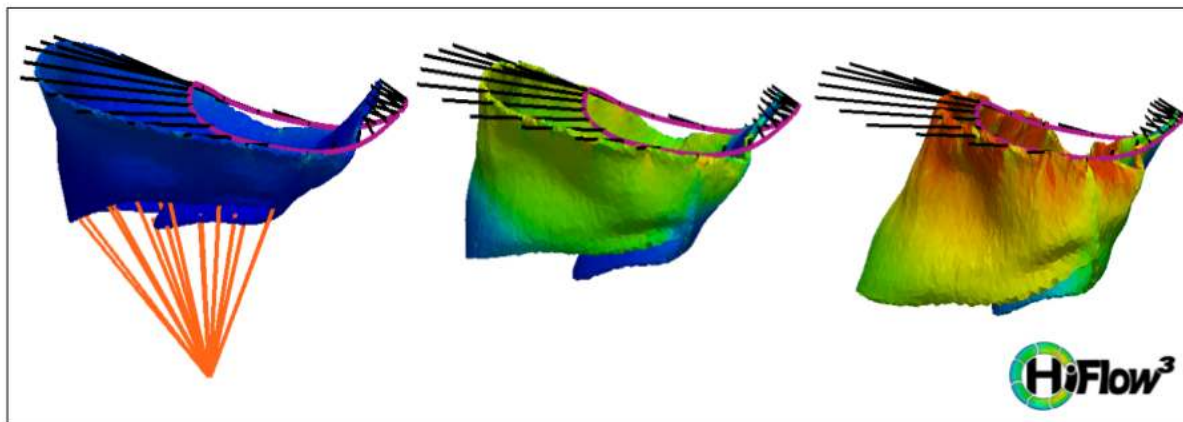


Figure 8.9: MVR simulation results: Visualization of the leaflet stretch.

Yet, interestingly, it has been shown by Chen et al. [CMMN04], that the anterior MV leaflet stretches during systole, which in turn is very well reflected by linear elasticity and the hence deformation-induced appearance of ghost forces, see Figure 8.9. All these issues again demonstrate the importance of our model visualization as in Figure 8.6 in order for the surgeon to have an understandable and transparent access to the underlying data and model features.

Also, once again, we emphasize that *clinical evaluation* has not yet taken place, as the focus of this work was put on the general feasibility of the intelligent integration of a biomechanical simulation into a cognition-guided cardiac surgery assistance system. Consequently, the simulated MV behavior was not compared with imaging-based observations or intraoperative measurements. However, it has to be pointed out that – knowing that data and model are not yet 100 % comprehensive and completely mature – the overall morphological deformation behavior and the von Mises stress distributions qualitatively agree with human observation and experience, e.g., in the Department of Cardiac Surgery at the University Hospital of Heidelberg. Moreover, they correspond to pertinent literature, such as the works of Choi et al. [CRMK14], of Stevanella et al. [SMC⁺11], or of Pouch et al. [PXY⁺12].

Yet, it has to be pointed out that, in addition to what these works presented earlier, through the overall cognition-guided setup and integration of our simulation and the above embedded pipeline, we here allow for *automatically set-up, knowledge-based, patient-individual MVR surgery simulation scenarios*.

This being said, in the future we plan to further develop and enhance our simulation setup, and to evaluate it in a clinical environment, including publication of the corresponding results in a dedicated work. For more details on the performance of our HiFlow³-based MVR simulation application, we refer to the following Section 8.2.

Semantic Data Infrastructure, Cognitive Tools Pipeline and CoSA Evaluation. Once the simulation is finished, the simulation results are semantically annotated by means of an attached RDF file and uploaded via the Cognitive Application interface to the central semantic XNAT file storage system.

The medical information of a set of test patients is thus comprehensively processed and enhanced through the entire pipeline – as shown above for one exemplary test case. This indicates the *successful integration* of our biomechanical FEM-based MVR surgery simulation application into the overall cognition-guided cardiac surgery assistance system.

Through execution of the entire pipeline it has been demonstrated that the whole setup and all of its individual components are *compatible* with each other, and that – given suitable initial input as described in Section 7.3 – the input/output data from these components is associated with each other *fully automatically* and *without errors*. We refer to Figure 7.8 again, for an illustration of the entire MVR surgery assistance workflow.

Beyond that, we have shown how our pipeline of dedicated MVR-related cognitive software tools seamlessly integrates into the joint overall *semantic surgery assistance system architecture* of our collaborative research center *Cognition-Guided Surgery* (SFB TRR 125). It has been demonstrated how it takes advantage of the underlying *semantic data infrastructure* and how it exploits the common *knowledge base*, e.g., for the selection of a patient-individually suitable annuloplasty ring prosthesis, or for representing and exchanging patient information.

The combination of the presented *data-driven pipeline architecture*, *cognitive software components* and an underlying *semantic data infrastructure* thus enabled the setup of a *prototypic simulation-enhanced cardiac surgery assistance system*. We have shown that this prototypic setup allows to consolidate and leverage heterogeneous medical information and surgical expert knowledge fully automatically, in order to, e.g., infer new simulation-based knowledge for MVR surgery support. It thus enables a more comprehensive and holistic surgical diagnosis and treatment planning. The cognitive tools pipeline-based *evolution of medical data* has been illustrated with respect to an imaginary MVR test patient in Figure 7.11.

As has been remarked above, the overall simulation-enhanced MVR surgery assistance prototype was not yet set-up and employed in a clinical testing environment, and a *clinical validation* has therefore not yet taken place – neither of the entire system, nor of its partial components. This being said, it is important to emphasize that several components of the overall system will first need to be further enhanced, improved and extended before a clinical validation makes sense. These comprise, e.g., with respect to image segmentation, a full ventricle segmentation algorithm in order to also depict the behavior of the papillary muscles and the chordae, or, concerning the biomechanical model, the patient-specific consideration of blood flow profiles, numbers of ruptured chordae, or material properties.

However, this is not part of this research work anymore, as the focus here was put on the successful integration of a biomechanical FEM-based MVR surgery simulation into an overall cognition-guided cardiac surgery assistance system.

For more details on the single steps in the overall simulation-enhanced MVR surgery assistance workflow, we refer back to Section 7.3.

Concluding, we remark that the here-above presented simulation-enhanced cardiac surgery assistance setup was deployed and presented in the context of the DFG assessment of the sub-project I03 *Functional Modeling and Numerical Simulation* of the SFB TRR 125 *Cognition-Guided Surgery* in January 2016. It has been demonstrated how heterogeneous medical data and surgical knowledge can be processed by means of a dedicated set of cognitive tools and

on the basis of a semantic data infrastructure in order to allow for comprehensive, knowledge-based, patient-specific FEM-based MVR simulation scenarios which may support cardiac surgery. The sub-project I03 thus obtained the only overall excellent grading, and got highly praised for its commendable realization of the *cognition-guidance concept* for surgery assistance.

8.2 Numerical Analysis and Performance Evaluation of the MVR Simulation

Having looked at the overall simulation-enhanced MVR surgery assistance system setup in the preceding section, and having evaluated the overall system's functionality, efficacy and usability, we will, now, investigate on the numerics and performance of the simulation component. Parts of the results that are going to be presented in the following have been published in our work [SES⁺15].

All computations were carried out on the *bwUniCluster*¹, which is located at the Karlsruhe Institute of Technology (KIT). We used up to 128 cores, distributed over up to 16 Intel Xeon compute nodes. Each of these nodes contains two Octa-core Intel Xeon processors E5-2670 (Sandy Bridge) which run at a clock speed of 2.6 GHz and have 8x256 KB of level 2 cache and 20 MB level 3 cache. Each node has 64 GB of main memory, local disks with 2 TB and an adapter to connect to the InfiniBand 4X FDR interconnect. For more information on the hardware architecture, we refer to the *bwUniCluster* online presence². The operating system is Red Hat Enterprise Linux (RHEL) 6.5. On top of this operating system a set of software components have been installed, like, e.g., the batch system MOAB, so users can – once their own software applications (as, for instance, our MVR soft tissue simulation) are deployed – submit their jobs in order for these to be executed when the required resources become available.

For the performance evaluation, we adduce the previously mentioned set of test scenarios for which our HiFlow³-based MVR soft tissue simulation was executed with respectively patient-specific MV meshes and patient-individually selected annuloplasty rings. The MV mesh representation was obtained using our MSML-based MVR simulation preprocessing workflow and the ring prostheses were chosen using the above depicted MVR deductive system for ring selection.

Limited by the ultrasound imaging **resolution** and the segmentation **quality**, the MSML-produced MV mesh representations consist in average of 20 000 to 23 000 *nodes*, which amounts to between 110 000 and 120 000 tetrahedral *cells*. When using Lagrange type FEs with a polynomial degree of 1, i.e., linear **Tet4** elements, this corresponds to a total of approximately 60 000 to 70 000 DoFs; using quadratic **Tet10** elements of polynomial degree 2, this corresponds to roughly 360 000 to 420 000 DoFs. Additional *refinement* is not reasonable, since no additional **accuracy** (in comparison to the ultrasound imaging resolution) is gained, neither does it yield better results in terms of a more realistic linear elasticity-based tissue behavior.

¹*bwUniCluster*: see <http://www.bwhpc-c5.de/wiki/index.php/Category:BwUniCluster>

²*bwUniCluster* hardware architecture: see http://www.bwhpc-c5.de/wiki/index.php/BwUniCluster_Hardware_and_Architecture

Due to the MV's strongly hydrated nature, one will need to use **almost incompressible materials**. For the patients in our test scenarios the MV material properties are accordingly determined by the *tissue density* $\rho = 1.04 \frac{\text{g}}{\text{mL}}$, and the two *Lamé constants* $\lambda \in \{28.46 \dots 68.64\}$ MPa and $\mu \in \{0.70 \dots 1.40\}$ MPa, the latter two of which correspond to a Poisson ratio $\nu = 0.488$ and a Young's modulus $E \in \{2.35 \dots 6.20\}$ MPa. Given the almost *incompressible tissue* and the *extremely thin leaflet morphology* (with an average thickness of 1.2 mm [MVG⁺12]), we make use of a polynomial FE degree of 2, since this is reported to yield more realistic results for linear elasticity than an FE degree of 1, and, in particular, strongly increases the simulated tissue flexibility [CK92, BLME13, Suw14]. Opposed to this, using conforming FEs with low FE degrees for simulating thin and incompressible materials reveals an excessive tissue stiffness which is known as *numerical locking* [Bra07]. From the mathematical point of view, this effect comes from the high ratio of the Lamé constants $\lambda \gg \mu$ due to the almost incompressible material properties. It is thus characterized by an increase of the quotient $\frac{C}{\alpha}$ in Cea's Lemma for $t \rightarrow 0$, and by the non-uniform convergence of the FE solution towards the real solution for $h \rightarrow 0$. Yet, as according to Cea's Lemma the quotient $\frac{C}{\alpha}$ directly enters the error estimation, one may expect errors that are remarkably greater than the approximation errors [Bra07].

As has been derived in Section 4.3, the numerical solution of the instationary contact problem of linear elasticity (eq. 4.104), which allows to model the MV behavior as illustrated in Figure 4.11, not only requires discretization in space (using the FEM approach) but also discretization in time, and hence a method for *time integration* to solve the system (eq. 4.113). We test the *explicit RK4 method* and the *implicit Newmark scheme*; see Section 4.3.2 or the book of Bathe [Bat96] for more details.

First, we employ the **explicit RK4 time integration scheme** with adaptive time stepping; see Section 4.3.2.1. However, as expected for stiff dynamic systems, it results in very small time steps [Wri06]. Right at the beginning of the simulation, i.e., during the annuloplasty ring-induced displacement part of the simulation, where there is no blood pressure yet, the time step size drops to values in the range from 1E-9 to 1E-8. Hence, even with massive parallelization, long computation times and large numbers of accepted time steps would yield hardly visible deformations only. Also, the step size must be expected to drop even farther to much smaller values as soon as blood pressure-induced shock waves come up and for leaflet penetration during the MV closing period. The MV leaflets then begin to oscillate in the force field and get more and more accelerated. Yet, the faster the particles move, the smaller the step size has to be in order for still accurate results. In our test runs, we terminated the simulation due to these observations.

The explicit RK4 scheme hence seems unsuitable for MVR soft tissue simulation, and especially for the simulation of contact between the two MV leaflets. We therefore apply the **implicit Newmark scheme**; see Section 4.3.2.2. Among many other implicit time integration schemes, such as the Crank-Nicolson method or the implicit Euler method, for linear elasticity problems in structural dynamics the implicit Newmark time integration scheme proved most stable and efficient [Bat96, EGDM15]. The two parameters β and δ allow to regulate its integration accuracy and stability. Moreover, it is particularly robust with respect to oscillations, and well-established for applications with Rayleigh Damping. Stability is given even for relatively large time steps $\Delta t \approx 1\text{E-}2$, however, only as long as the elastic response of the structure is

Table 8.1: Run-time performance of our HiFlow³-based MVR simulation using the Symmetric Gauss-Seidel preconditioned CG solver, for an MV mesh with a total of 113 513 linear **Tet4** elements and $N = 63\,447$ DoFs.

Number of parallel processes	1	2	4	8	16	32	64	128
Number of nodes \times cores	1×1	1×2	1×4	1×8	2×8	4×8	8×8	16×8
Run-time (sec)	8086	4102	1997	1104	603	365	235	161

governed by the low frequency modes, i.e., when there is no interpenetration of the two leaflets which shock-push the tissue. Remarkably, the implicit Newmark scheme with time step sizes of approx. $1\text{E-}2$ it still works fine with the pulsatile hemodynamic loading during the cardiac cycle, as long as there is no leaflet contact. Yet, as soon as the two leaflets approach each other too much, to under the defined minimal gap distance d_{gap} of $\varepsilon_{\text{tol}} = 1.5$ mm, and hence, as soon as the penalty term p_C potentially jump-grows towards infinity, we have to strongly decrease the time step size. With time steps between $1\text{E-}3$ and $1\text{E-}2$, we can catch the event of contact, i.e., avoid the occurrence of leaflet interpenetration, and guarantee for an almost gentle increase of the penalty term, and hence for simulation stability.

Hence, we clearly opt for the implicit Newmark time integration method, to efficiently obtain stable and accurate simulation results in still adequate simulation times. The system matrices and vectors are then assembled according to the algorithm shown in Section 4.3.2.2.

As can be inferred from the assembly equations (eq. 4.109) and (eq. 4.110), and from the Newmark algorithm in Section 4.3.2.2, one obtains a *symmetric, positive definite, sparse system matrix* in every Newmark time step. Therefore, to solve the thus arising linear systems, we apply a **Symmetric Gauss-Seidel** or **Algebraic MultiGrid (AMG)** preconditioned **CG solver** in every Newmark step. For more information and theory on the solver and preconditioner selection, see the works of Saad [Saa03] and Meister [Mei15].

Given a thus typical problem size of approximately $N = 65\,000$ DoFs for an MV mesh with linear **Tet4** elements, the Symmetric Gauss-Seidel preconditioned CG method shows very nice **convergence properties** and low **iteration numbers** between 110 and 230 (for increasing numbers of parallel processes) for a tolerance threshold of $1\text{E-}8$ for the norm of the residual; compare Table 8.1 and Figure 8.10.

Consequently, we can compute the **speedup** of a parallel execution

$$S(p, N) = \frac{T(1, N)}{T(p, N)} \quad (8.1)$$

with the run-time $T(p, N)$ for an execution with p parallel processes and problem size N , which translates into the **efficiency**

$$E(p, N) = \frac{S(p, N)}{p} \quad (8.2)$$

of between 98.3 % (for lower numbers of parallel processes) and 39.2 % (for higher numbers of parallel processes); see Figure 8.11. We emphasize that – as can be seen from Table 8.1 – the computation time for an entire MVR simulation scenario as described in the simulation part of Section 8.1 takes less than 3 minutes when executed on 128 processes. Please note, that – in

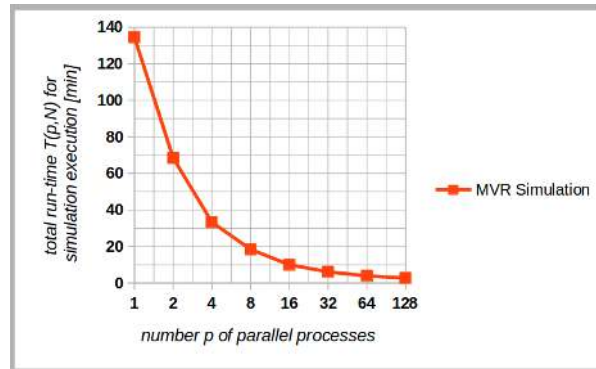


Figure 8.10: Run-time performance of our HiFlow³-based MVR simulation using the Symmetric Gauss-Seidel preconditioned CG solver, for an MV mesh with a total of 113513 linear Tet4 elements and $N = 63447$ DoFs; see Table 8.1 for the exact numbers.

addition to this – several MSML-controlled annuloplasty ring-specific MVR simulation scenarios may run in parallel when more resources are available; see again, Section 5.2.2, paragraph 4.

In spite of the overall nice performance results in Table 8.1, and in Figures 8.10 and 8.11, respectively, one still observes that the algorithm’s efficiency degrades with increasing numbers of parallel processes. As mentioned before in the third paragraph in Section 5.2.2, this is mainly due to two aspects: Firstly, when solving fully coupled linear systems, information has to be communicated and synchronized in every preconditioner/solver iteration. However, assuming a constant problem size, when parallelizing and distributing the computation load over several processes, the computation to communication ratio will decrease, and hence make the algorithm less efficient. Moreover, looking at the morphology of an MV mesh, the bad volume to surface ratio will even negatively amplify this effect. Secondly, due to the application of the block-Jacobi type of Symmetric Gauss-Seidel preconditioner, couplings from different (distributed) subdomains are neglected in the preconditioning step, such that the preconditioning effect becomes less effective with an increasing number of subdomains, i.e., with more massive parallelization. Consequently, the CG solver has to perform more iterations to reach a given tolerance threshold, which becomes apparent when iteration numbers rise from 110 to 230 (and possibly even beyond) for respectively higher numbers of processes. Of course, this causes an increased need for computation time, too.

Accordingly, one might reason that a multigrid preconditioner, such as the Boomer AMG (Algebraic MultiGrid) preconditioner from Hypre [FJY06], would overcome this problem, and hence allow for better performance. Looking at the iteration numbers of an algorithm setup that builds on the combination of the *AMG preconditioner* and the *CG solver*, we correspondingly found a relatively low number of 14 to 17 iterations, which remains relatively constant even for rising numbers of parallel processes. However, comparing the run-time results of such an AMG plus CG setup with the afore-listed results of a Symmetric Gauss-Seidel plus CG combination, one will see that the AMG preconditioned CG method performs *worse* in terms of the overall simulation time than the Symmetric Gauss-Seidel preconditioned CG method, which is due to the still relatively small global number of DoFs in the presented common test scenarios, and due to the fact that the AMG iterations are by far more compute- and time-intensive. Yet, as

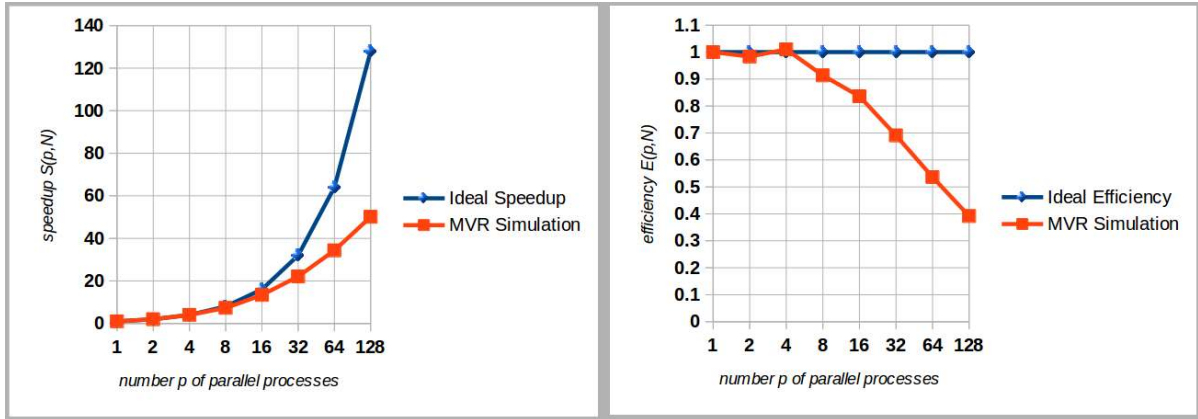


Figure 8.11: Speedup (*left*) and efficiency (*right*) of our HiFlow³-based MVR simulation using the Symmetric Gauss-Seidel preconditioned CG solver, for an MV mesh with a total of 113 513 linear Tet4 elements and $N = 63\,447$ DoFs.

the number of iterations is not expected to remarkably grow for larger problem sizes either, it can be assumed that the AMG / CG combination will outperform the Symmetric Gauss-Seidel / CG combination as from a certain larger problem size on and beyond.

Hence, even though we observe a relatively bad overall performance of the AMG preconditioned CG algorithm in comparison to the Symmetric Gauss-Seidel preconditioned CG algorithm for our presented small test scenarios, it is to be pointed out that having implemented an easily controllable `xml` input interface for the Boomer AMG preconditioner from Hypre, we do allow for larger systems to be solved with our simulation algorithm, too. These might, for instance, arise once higher resolved imaging and segmentation techniques are achieved in the clinic environment, such that more accurate, i.e., higher refined, simulations make sense, too.

This being said, we again emphasize the importance of an *optimized contact search algorithm*, as presented in the second paragraph in Section 5.2.2. A large number of MV leaflet surface elements need to be checked for potential contact in every simulation time step, and carrying out these checks clearly dominates the overall time of the respective step and hence of the entire simulation. We have worked towards maximally exploiting the potential of parallelization in this context, and hence minimizing the respective computation time.

Concluding, in this Section, we have analyzed the simulation performance and found that the overall **run-time performance**, **speedup** and **efficiency** of our simulation algorithm is very satisfactory, and that the respective features correspond to those of comparable other mature FEM simulation softwares and applications as mentioned in Chapter 3.

Lastly, it is to be remarked that for the presented evaluation setup, there is **no analytical or reference solution** available. However, our partners in the Department of Cardiac Surgery at the University Hospital of Heidelberg generally agree with the virtually simulated MV behavior based on visual assessment. Thorough evaluation and comparison with an accepted gold standard is, however, imperatively needed once a more advanced model and simulation is

implemented in order to thus enable truly reliable simulation results and an application in the context of surgery assistance. This will be subject to future research and development work.

8.3 Outlook with respect to further Evaluation of Model and Simulation

Looking at the reliability of the obtained simulation results, we see a great potential for evaluation in the application of methods for **Uncertainty Quantification** (UQ). For instance, an evaluation of the simulation may be achieved through a setup which compares virtually and experimentally obtained post-surgery results with respect to the sizes of *post-MVR coaptation areas* throughout the cardiac cycle. On the one hand, our clinic partners have developed algorithms aiming for the automatic assessment of the MV coaptation zone from 4D ultrasound images [ELAM⁺15]. These can also be applied post-surgically to thus assess the quality of the annuloplasty ring implantation by means of post-surgical coaptation quantification. On the other hand, we can compute the size and shape of the coaptation zone based on the above-described patient-specific MVR simulation scenarios. Adding on top of this the knowledge that certain material parameters or blood pressure profiles may vary to some extent, and describing this knowledge by means of probability distributions, with the application of methods for UQ, we may obtain reliable results which realistically represent the MV behavior and potential morphologic deviations. Both may be compared and hence hold for evaluation purposes.

Similarly, or on top of this, as has already been mentioned before in Section 3.2 (C), one may aim at enhancing and patient-specifically calibrating our simulation algorithm with methods from the class of **Data Assimilation** algorithms. This way, a rough, initial patient-specific simulation scenario setup can iteratively be further calibrated according to experimental data observations. A usual *once set-up, then static* simulation scenario setup could thus be gradually adapted and become even more patient-specific and realistic.

Lastly, looking at the overall good simulation performance, one may reckon that further performance improvement can be achieved, e.g., through exploitation of the potential of **GPU-accelerated computations**. Initial tests with the presented HiFlow³-based MVR soft tissue simulation algorithm have already been conducted: Based on an earlier work by Anzt et al. [AHR10], who implemented a *CUDA-based efficient CG solver* for the FEM software toolkit HiFlow³ to solve large linear systems using GPUs, we extended the respective data structures in our MVR soft tissue simulation algorithm in order to thus facilitate the application of this CUDA-implemented CG solver for the above derived linear systems. However, it quickly became clear that the hence set-up hybrid CPU-GPU-based simulation computations performed worse than CPU-only computations. This is mainly due to the fact that the theoretical performance advantage of the GPU-based CG solver routine cannot outperform the disadvantageous, additional simulation time overhead caused by the memory transfer between host and device, i.e., between CPU and GPU, which is imperative before and after every solution step as system matrices and vectors get newly re-assembled in every time step. Moreover, the performance gain of the GPU-based CG solver is not yet fully exploited, since a GPU-based implementation of a HiFlow³-compatible preconditioner is so far missing, such that the GPU-based *unpreconditioned* CG solver must compete with a CPU-based *preconditioned* CG solver. Both entails a promising potential for further developments and for an improved performance.

9 Summary, Discussion and Outlook

Having evaluated the obtained results and the overall setup and performance of our cognition-guided, patient-individual, simulation-enhanced cardiac surgery assistance prototype in the preceding Chapter 8, in the following, we will give a brief summary of the entire PhD work in Section 9.1, and conclude with a final discussion in Section 9.2. Lastly, we suggest an outlook to future further work in Section 9.3.

9.1 Summary and List of own Contributions

In this work, a prototypic setup of a *cognition-guided, patient-individual, simulation-enhanced cardiac surgery assistance system* has been presented.

We have motivated our work by the example of *MVR surgery* in Chapter 2. The valvular apparatus and its functionality have been described, and it has been pointed out which pathologic circumstances require for surgical treatment by means of MVR. When dealing with such complex surgery, patient-specific and surgical expert knowledge-based surgery planning is of utmost importance. The possibility not only to plan, but to simulate, too, some steps of the intervention hence may be extremely beneficial for surgeons in order for them to forecast relevant clinical situations and the post-surgical MV behavior.

However, it is remarkably difficult to provide surgeons with patient-specific, situation-adaptive and knowledge-based simulation tools that they could actually handle in order to produce surgically beneficial results. In Chapter 3, we investigated on the *state-of-the-art* in the field of simulation-based cardiac surgery assistance in particular and of simulation-based surgery support systems in general, and pointed out *current open challenges* with respect to biomechanical simulation, cognition-guidance and medical information and knowledge processing.

We found that there is an urgent need for a *cognition-guided, simulation-enhanced surgery assistance* which – just like a human surgical assistant – supports cardiac surgery by providing the operating surgeon *fully-automatically* with patient-specific, situation-adapted, surgical expert knowledge-based, comprehensive MVR surgery simulation scenario results. These surgery simulation scenarios and their underlying modeling needs to be based on a preceding holistic analysis of the available patient data and on the consideration and processing of surgical knowledge.

Therefore, we proposed a simulation-enhanced MVR surgery assistance system setup, which integrates a knowledge-based, patient-specific biomechanical MVR surgery simulation, and thus provides surgeons with useful additional simulation-based information.

In Chapter 4, we described the *mathematical MVR model* behind our biomechanics-based *FEM simulation* of the MV behavior. The *implementation* of our HiFlow³ FEM simulation, as well as the application's control interfaces and performance features are then detailed in

Chapter 5. Subsequently, in Chapter 6, we present the *Medical Simulation Markup Language* (MSML), and how it simplifies the entire biomechanical modeling workflow which starts with image segmentation and goes via meshing to modeling and simulation. Thereon-based, we dedicate ourselves to the comprehensive, fully-automated information preprocessing in order to obtain patient-specific MVR simulation scenarios. In Chapter 7, we then present the underlying *cognitive system architecture* and the *semantic data infrastructure* as the basis of our collaborative research center *Cognition-Guided Surgery* (SFB TRR 125), as well as the *cognitive software setup* which finally facilitates the combination of all tools and applications involved in the surgery assistance workflow in order for the holistic information processing of heterogeneous medical patient data and of surgical knowledge. Looking at the specific case of MVR surgery, we suggest in Section 7.3 a comprehensive pipeline of *cognitive tools for simulation-enhanced cardiac surgery assistance*, which is evaluated in terms of its overall functionality and performance in Chapter 8.

With respect to the state-of-the-art, this work has made the following *major contributions to cognition-guided, simulation-enhanced cardiac surgery assistance*, published in the respectively referred conference proceedings or journal papers:

- A *concept* has been developed, which allows to integrate a patient-individual, biomechanical FEM-based MVR simulation into a cognition-guided surgery assistance system; see our interdisciplinary work [SEZ⁺15].
- The central *FEM soft tissue simulation application* builds upon a simplified, but generic and flexibly extensible mathematical MV model, which allows to simulate the patient-specific post-MVR-surgery behavior of the MV under consideration of a respectively implanted annuloplasty ring prosthesis; see [SES⁺15].
- The *implementation of the MVR surgery simulation application* aims at exploiting the available High-Performance Computing infrastructure, and is designed to be stable, efficient and reliable, with competent and flexible control mechanisms and interfaces in order to facilitate the consideration of patient-specific medical data and of surgical expert knowledge; see [SES⁺15].
- To facilitate and simplify the overall biomechanical modeling workflow, from image segmentation via modeling and simulation to postprocessing, the *Medical Simulation Markup Language* (MSML) has been proposed; see our joint across-project work [SSS⁺14].
- Specifically, for the clinical MVR surgery assistance use case, a *comprehensive MVR information preprocessing pipeline* has been designed and implemented on the basis of the MSML. It allows to fully-automatically set up MVR simulation scenarios that are both patient-specific and comprehensive as well as based on surgical expert knowledge; see the works [SKS⁺16] and [SKS⁺15].
- In order to have the system intelligently process medical patient data and surgical information and knowledge, a *cognitive surgery assistance system architecture* has been proposed as a joint research and development effort in the framework of the collaborative

research center *Cognition-Guided Surgery* (SFB TRR 125). In particular, this architecture includes a *semantic data infrastructure* and *cognitive software components* which can be composed to complex, comprehensive, data-driven information processing pipelines. These pipelines benefit from and contribute to the common SFB knowledge base, and coordinate all required tasks for machine-based MVR surgery assistance: from data digitalization and semantification, via segmentation and surgical reasoning, to simulation preprocessing and simulation management; see [FMK⁺16] and [SPW⁺16].

- The common *SFB knowledge base* facilitates both the semantic storage of medical patient data and surgical knowledge, as well as the management of cognitive tools and algorithms which work on this data and information. It thus allows that heterogeneous patient-individual data can be put in relationship with comprehensive surgical knowledge which is then further processed; see [FMK⁺16].
- For the specific clinical use case of MVR surgery, a data-driven *pipeline of cognitive tools for simulation-enhanced MVR surgery assistance* has been set-up and deployed. Heterogeneous patient-specific data (images and parameter data) is processed under consideration of surgical expert knowledge, and thus allows for an evolution and enhancement of medical data, meaning that surgeons are provided with additional simulation-based information in order for a more comprehensive and holistic surgical treatment planning; see [SEZ⁺15] and [SPW⁺16].
- Finally, an outlook *towards semantic surgery simulation* is proposed, and a preliminary, simple ontology-based system setup is presented, aiming at the transfer of the MVR-related surgery assistance setup to arbitrary other clinical use cases; see [SSSVH16].

The overall system hence combines *numerical soft tissue simulation* and methods from the field of *data science* and *information processing*, and thus goes a step *towards cognition-guided, patient-specific numerical simulation for cardiac surgery assistance*. To the best of the author's knowledge, no such system has ever been proposed and prototypically implemented before. The respective system components have been presented and published in journals and on conferences, and the implementation is partly available open-source on GitHub and via Surgipedia, or upon request to the author.

Once the system's prototypic components are more advanced and collectively operational, great *clinical benefit can be expected* thanks to the cognition-guidance approach, which actually allows that the available amounts of medical patient data and of surgical expert knowledge can *automatically* and *holistically* be processed in order for more comprehensive, knowledge-based, patient-individual surgery simulation scenarios.

The presented research and the respective prototypic implementation are hence to encourage further research efforts and development works in order to further develop and improve the applicability and usability of simulation-enhanced surgery assistance. Several items for future work towards improvement and optimization have been suggested in the outlook sections of the above chapters, e.g., in the direction of more advanced biomechanical models, or of full automation to go towards a complete usability.

9.2 Discussion and Conclusion

Having summarized the results of this PhD work in the preceding section, now, we will conclude with a final discussion, both with respect to the novelty, contribution and placement of the underlying scientific work, and also with respect to the applicability and usability of the resulting system architecture and the comprised software components in a clinical environment.

A *rationale* for the application of *FEM-based MVR simulation* in MVR surgery assistance has been given in Chapter 2, and the benefits for preoperative planning are manifold: Surgeons and staff in the OR can obtain a patient-individual model of the pathological MV behavior preoperatively, and therefrom generate simulation scenarios which perform a variety of potentially suitable surgical MVR strategies. This way, the surgical team can determine, before ever entering the OR, the morphologically and biomechanically superior MVR strategy for the respective patient – both with respect to the deformation behavior and the coaptation functionality, and also with respect to the expectable stress distribution in the MV leaflets which may affect a future tissue failure.

In the process of modeling, the developed model was inspired by existing state-of-the-art works and research projects, as described in Chapters 3, 4 and 5. However, as motivated in Section 4.2, the goal of this work does not consist of the development of a more advanced and more comprehensive biomechanical model, but rather of the setup of a simplified, but generic and flexibly extensible model with competent and flexible interfaces which allow to account for all important model feature types and for all relevant impact factors that determine the MV behavior. After all, any employed model is always incomplete with respect to specific features and makes use of *simplifying assumptions*. In our case, for instance, the papillary muscles and chordae tendineae are not captured by means of the image segmentation tool and their effects are therefore not properly reproduced by the simulation tool either. In turn, this directly entails a *limited reliability* of the simulation results.

Besides the *limited model completeness*, the second major limitation that has prevented the widespread adoption of FEM-based simulation as a surgical tool lies in the fact that an FEM-based surgery simulation is only then really beneficial and useful for the surgeon, if it considers and accounts for current surgical expert knowledge and if it integrates all available medical patient data – just as a human surgeon would, too. In the proposed work, we accordingly make an important step towards achieving this requirement: As described in Chapters 6 and 7, we enable the holistic analysis, interpretation and the processing of *heterogeneous medical patient data* (both images and parameter data) by means of a dedicated set of cognitive information processing tools, and thus also exploit the available *surgical expert knowledge*, in order to set up MVR surgery simulation scenarios which are both *patient-specific* and *comprehensive* with respect to a reasonably mature underlying biomechanical model. Moreover, through interconnecting the respective tools via a complex, cognitive, data-driven information processing pipeline, we facilitate the *full automation* of the entire simulation-enhanced surgery support workflow.

Hence, based on this research work and on the developed prototypic system setup, we are able to – for the first time – fully-automatically and comprehensively analyze, interpret and process relevant patient-specific medical data under consideration of surgical expert knowledge in order for *cognition-guided, simulation-enhanced cardiac surgery assistance*. In the future, once such a system setup is more advanced, e.g., in terms of the above mentioned assumptions

and limitations, such cognition-guided, patient-individual FEM surgery simulations have the potential to become a surgeon's best friend, arming them with enhanced knowledge about potentially best-suitable MVR operation strategies for their respective patients.

We refer back to Chapter 8, where the proposed cognitive simulation-based cardiac surgery assistance setup was thoroughly evaluated, and where it was shown that the aforesaid challenges could successfully be coped with. The simulation results and the overall system output did indicate that first, auspicious steps towards cognition-guided, patient-specific numerical simulation for cardiac surgery assistance have been achieved. However, *clinical evaluation or validation* has not yet taken place, so far, neither of the entire system setup, nor of the individual components. As has been motivated above, several components – from image segmentation to modeling and simulation – first need to be further improved and advanced before a clinical validation of the overall system will make sense.

Yet, the presented simulation-based MVR assistance workflow is designed in a way such that the single *workflow steps integrate well into the standard clinic environment and into the traditional surgical treatment workflow*. This is accomplished thanks to the fact that (a) on the one hand, 3D images of the valvular apparatus are acquired anyways, and clinically relevant patient parameters are recorded, in any case, too, and (b) on the other hand, with our cognitive, data-driven pipeline, patient data and surgical information is automatically further processed (in the background) in order for the simulation as soon as it is available.

The simulation-based system thus *smoothly* enhances the surgeon's 3D perception through generating a 3D representation of the MV morphology and through simulating the post-MVR-surgery behavior with respect to patient-specifically selected annuloplasty ring prostheses. Also, the fact that useful additional simulation-based knowledge is provided to the surgeon without any manual intervention and without any human work overhead prominently indicates the system's *usability* and *robustness*, as well as its *applicability* in the OR. Lastly, the system's overall *efficacy* will likely even become more apparent once the suggested further enhancements and extensions (as in the discussion parts of Chapters 4 to 7) have been accomplished, such that an even more fully comprehensive and reliable, patient-specific, knowledge-based surgery simulation is enabled.

At this point, it is to be emphasized again, that the overall simulation-enhanced cardiac surgery assistance setup, as described in Section 7.3 and employed in Chapter 8, was deployed and demonstrated for the first time in the context of the *DFG assessment* of the sub-project I03 *Functional Modeling and Numerical Simulation* of the SFB TRR 125 *Cognition-Guided Surgery* in January 2016. It has been demonstrated how *heterogeneous medical data and information* can be processed by means of a dedicated set of *cognitive tools* under consideration of *surgical expert knowledge* and on the basis of a *semantic data infrastructure* in order to facilitate the setup of comprehensive, cognition-guided, knowledge-based, patient-specific MVR surgery simulation scenarios for cardiac surgery support. The respective setup derives maximal benefit from using and feeding the common SFB knowledge base, and is thus capable of providing additional surgically valuable simulation-based information for surgical analysis, diagnosis and treatment planning to the surgeon. As the only sub-project, I03 thus obtained the overall grading **Excellent**, and got highly praised for its commendable prototypic realization of the *cognition-guidance concept* for surgery assistance.

Looking at the state-of-the-art in simulation-enhanced cardiac surgery assistance and at the respectively open research questions and challenges as listed in Chapter 3, one accordingly quickly figures that the overall idea, concept and implementation of the presented prototypic *cognition-guided, simulation-enhanced cardiac surgery assistance* is a prominent *innovative and unique feature of this work*, which thus combines the fields of cognition guidance, data science and numerical simulation in the context of cardiac surgery.

For the first time, an intelligently and patient-specifically set-up biomechanical FEM surgery simulation and its results have been connected with a comprehensive, semantic data management and information processing architecture, and integrated into a cognition-guided surgery assistance and decision support system.

The proposed simulation-enhanced surgery assistance workflow provides a means for virtually assessing *pre-operatively* the patient-specific *post-operative* behavior of the MV *after* annuloplasty ring implantation, *before* the actual surgery takes place. From this opportunity, valuable *clinical benefit* can be expected, as has been motivated in Chapter 2: Through anticipating potentially critical surgical outcomes, e.g., with respect to a postoperatively high stress distribution or an incompetent coaptation functionality, which may be due to the implantation of a suboptimal ring prosthesis, operation time and cost can be saved, and patients are subjected to fewer complications. In particular, one may expect a lower reoperation rate thanks to a more likely directly suitable selection of patient-individual ring prostheses.

Finally, it is to be remarked, that, in the presented cognition-guided, simulation-enhanced cardiac surgery assistance system, most of the different software components are custom-tailored for the specific use case of MVR surgery assistance. However, due to the generic setup of the overall system and its semantic architecture, the concept of cognition-guided, patient-specific, simulation-enhanced surgery assistance can be expected to be *transferable* to other fields of surgery, too. In particular, we have proposed in this context a first, prototypic development which goes towards *semantic surgery simulation* [SSSVH16].

As a final side note, the social relevance of the related research and the importance of the entire research collaboration on *Cognition-Guided Surgery* for healthcare and surgery has been brought into consciousness on the *Science Vision Conference*¹ in Karlsruhe, Germany, in 2012; see Figure 9.1:

The concept of *Science Vision* is to let *Design explain Science*, through having media communication researchers, natural scientists and engineers pair up, so they are able in a science communication way to make a broad audience aware of current socially relevant, cutting-edge research fields. Nicolai Schoch was invited to present his research work in the field of *Cognition-Guided Medical Simulations*, too. Together with design students from the Karlsruhe School of Arts (HfG), Germany, he outlined his work on the functional modeling and simulation of physiological processes in the human body by the example of simulation-based liver surgery assistance.

The talk was part of a farraginous series of presentations, all of which were to fascinate and inspire the audience, and which were to help foster public awareness and discussion. In particular, the presentation of Nicolai Schoch is available via the Website of the *Science Vision Conference* and on YouTube: <https://www.youtube.com/watch?v=PZhrDK4NXkc>

¹Science Vision Conference: <http://www.science-vision-conference.de/>

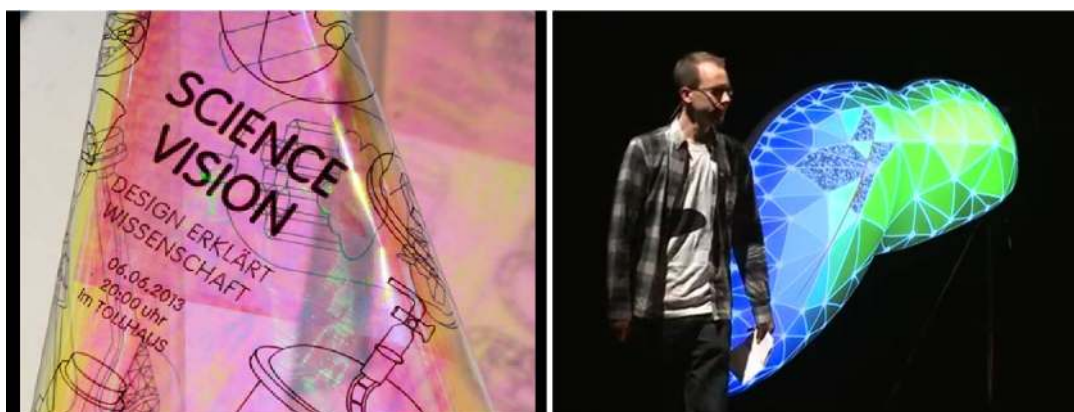


Figure 9.1: Science Vision Conference: the logo (*left*) and a photo taken during the talk (*right*).

9.3 Outlook and Future Work

In this last section, we suggest an outlook to future related work. On the one hand, we present two promising future applications that build upon the presented cognition-guided, patient-individual, simulation-enhanced cardiac surgery assistance system prototype; see Section 9.3.1. On the other hand, we list what we see as the most important enhancements and the most promising extensions in the context of further research on cognition-guided simulation assistance in surgery; see Section 9.3.2.

9.3.1 Possible Future Applications to Build Upon the Presented Work

Besides the above indicated idea of making use of the presented simulation-enhanced assistance prototype for the actual *surgical treatment planning*, where patient-individually selected annuloplasty ring prostheses can preoperatively be assessed, we see a great potential of our assistance workflow for *surgery training*, too:

Sturla et al. [SVO⁺15] use biomechanical simulations to evaluate the biomechanical drawbacks of different MV repair techniques for neochordal implantation, thereby aiming to help answer the question on when an apparently optimal repair can fail.

Similarly, future surgeons-to-be could exploit our proposed surgery simulation system for *trying out surgical MVR strategies*. If they are not sure whether ring type A or B, or ring size X or Y is more suitable for a respective MV, they can have several surgical scenarios performed and look at the virtual post-operative results. In this way, they may figure, that a smaller ring possibly stiffens the tissue too much around the annulus, or that a bigger ring will not allow for a satisfyingly large coaptation area, or that a saddle-shaped ring entails a more relaxed tissue than a flat ring would. In addition to this, we again want to emphasize the *importance of surgical experience* for the operation outcome as investigated on by Gammie et al. [GSG⁺09]. According to this study, the success of a surgical operation still largely depends on the experience of the operating surgeon, which could safely and risklessly be acquired through surgery simulation frameworks such as the proposed one.

In a first discussion with a cardiac surgeon in the context of the preparation for the DFG

assessment of the SFB TRR 125 project I03 in January 2016, we figured the great value of such *free trials*, i.e., of virtual surgery, and realized the surgical benefits that come from being able to virtually assess two or more invasive surgical strategies that are custom-tailored for a respective patient's pathology.

Another promising application that builds upon the proposed simulation system and that appears to be soon within reach, makes use of a technique known as *Rapid Prototyping* or *3D Printing*: This technique allows to produce real 3D objects from virtual 3D models, and could be applied to 3D models of *patient-individually optimized annuloplasty ring prostheses*.

Literature shows that first attempts and studies with this technique have already been conducted. Maisano et al. [MRS⁺05] performed an FEM simulation of a novel, dog-bone-shaped annuloplasty ring design, and they came to the conclusion that the use of disease-specific prostheses is necessary to improve the results of MVR surgery. Accordingly, it seems reasonable to generally *biomechanically test new ring designs* in a virtual environment *before production* by means of biomechanical MV simulations, and to thus confirm their efficacy and expediency.

Going further, Diaz Lantada and colleagues, suggested to *optimize size and shape* of annuloplasty rings with respect to goal functions that are integrated into the MV simulation and which maximize the coaptation zone or minimize stresses [DLVFM⁺10]. However, their approach has not yet been implemented in practice, mostly for the reason of design and manufacturing difficulties, as has been the case with other personalized implant and prosthetic developments, too. Similarly, Graser et al. [GWAM⁺14] suggest to use rapid-prototyping technologies to produce patient-specific rings from biocompatible titanium as evaluated by Suendermann et al. [SGP⁺13] in an animal study. Yet, Graser's model is based on shape priors and on several static state distance optimizations, whereas an FEM simulation-based optimization would allow for the simulation (and optimization) of the dynamic valve behavior throughout the entire cardiac cycle.

Using our fully automated, patient-specific MVR simulation setup, such results may be in reach, soon, such that both a patient-individual assessment of commercial ring designs, and a patient-specific shape/size optimization of personalized ring designs could be facilitated. The former of these two would check the mean closing functionality of the MV (for sets of typical patient classes) and calculate the upcoming stress distributions, when the MV has been subjected to the implantation of new ring shapes. The latter of the two would be able to virtually optimize 3D ring designs, which, in turn, are suitable for rapid prototypic, e.g., using the above-mentioned biocompatible titanium materials.

9.3.2 Possible Ongoing Research and Development

Lastly, we list what we regard to be important future enhancements and extensions in the context of further research and development in cognition-guided surgery simulation assistance.

Enhancement of the biomechanical model and Uncertainty Quantification. As has been repeatedly stated before, the employed biomechanical MV/MVR model and simulation setup is – compared to the state-of-the-art in the field of biomechanical MV modeling – rather simple, but generic and flexibly extensible, and it comes along with competent and flexible interfaces in order to facilitate the integration of patient data and surgical knowledge. The model does

account for the most important features of the MV and for all relevant impact factors that determine the MV behavior (such as blood pressure, chordae tendineae, annuloplasty, etc.), as described in Section 4.2. However, the representation of these features is strongly simplified, and allows for miscellaneous *model extensions and improvements*, e.g., towards nonlinear, transversely isotropic elasticity, multiply-layered MV tissues, secondary chordae, etc.; see again, Section 4.2.2. More advanced models are not only available in the literature, but have been used for numerical simulations as well; see again, the coverage of the state-of-the-art in Chapter 3.

This being said, whichever advanced and sophisticated model is used, there will always be specific features that are not properly captured or reproduced, which in turn entails a respectively limited reliability of the simulation results.

Knowing about this uncertainty, one can either make use of more advanced technologies (e.g., in image acquisition or in segmentation) in order to obtain more reliable and more comprehensive morphologic information about the MV (e.g., with respect to papillary muscles and chordae tendineae), or represent the given uncertainty by means of appropriate probability distributions or through dedicated methods for Uncertainty Quantification (UQ). We have accordingly emphasized the importance of UQ for medical simulations in Section 3.2 and in the discussion parts throughout this work.

In order for a simulation-enhanced surgery assistance system to possibly *go live*, we consider an enhanced biomechanical model as well as the integration of methods for UQ to be essential.

Performance improvement and exploitation of HPC technology. More advanced models and the integration of methods for UQ bring along increased simulation algorithm complexity and higher computational costs. To cope with these, the potential of the available HPC infrastructure – above all of (GP)GPU clusters – needs to be further exploited. As mentioned in Section 8.3, first work towards the achievement of a GPU-accelerated simulation has already been made, and promising further work will primarily consist in the implementation of a HiFlow³-compatible GPU-based preconditioner in order to enhance the GPU-based CG solver.

Biomechanical Modeling and Surgery Simulation Ontology. In order to provide the surgeon with reliable simulation results, not only the results themselves need to be computed with high accuracy and to the farthest possible exclusion or quantification of uncertainties by means of methods of UQ. In fact, it must be clear and commonly agreed upon, what input/output notions are referred to and what measurements are applied for their computation.

As motivated in Section 7.4, a suitable biomechanical modeling and simulation ontology can be expected to satisfy this requirement. On the one hand, such an ontology is to collect and interlink the comprehensive knowledge about numerics, modeling and simulation. On the other hand, it is to semantically represent this knowledge, and to abstract and generalize it, so it can compatibly be connected with other semantic annotations and ontologies, most importantly an upper ontology (like the BFO) and a surgical/anatomical ontology (like the FMA).

This way, ideas, notions, simulation setup descriptions, and simulation scenario results can be shared within a hopefully growing surgery simulation community, and commonly be used among surgeons across different clinical departments or institutions, since all items are represented with respect to one common vocabulary and measurement collection.

A prototypic development going towards semantic surgery simulation has been suggested

in Section 7.4 and presented at the *Surgical Data Science* Workshop in 2016 [SSSVH16]: The proposed system prototype encompasses a structured, semantic representation of modeling and simulation knowledge and an underlying numerics properties decision tree, which is linked to a corresponding semantic representation of anatomical and surgical properties, so the system is enabled to deduce simple, patient-specific simulation scenarios. Hence, the usage of a surgery simulation ontology and a system as the above proposed one will certainly serve for integration purposes and foster in-OR-usability and acceptance of surgery simulations.

Lastly, added value of employing an ontology is also to be expected in an easier transferability of the prototypic developments and of knowledge and processes to other application fields and use cases. Moreover, predictions are made possible, which in turn opens up new opportunities for semantics-based machine learning algorithms.

Combination of FEM surgery simulation and Data Assimilation methods. Using the proposed simulation system prototype along with the comprehensive simulation preprocessing workflow and the underlying semantic data infrastructure, we enable patient-individual medical data and information to be accounted for. The setup of the simulation scenario – once it is *thus* set up – then *is* patient-specific, but *static* and assumed to be correct for the complete simulation run. Variations of the boundary conditions, slight adaptations of model parameters, new impact factors, or new additional knowledge, cannot be included into the simulation setup anymore. Consequently, image data often reveals a tissue behavior which is different from the simulated behavior.

Methods for *Data Assimilation* (DA) allow to exploit this new or additional information and to continuously merge it into the simulation algorithm in order to hence facilitate a further, iterative, patient-specific calibration of the model and simulation setup. An initial static patient-specific simulation scenario setup (as accomplished by the current system prototype) may thus *iteratively* be further adapted and calibrated according to experimental data observations [PK15].

Hence, on the way *towards a more complete Cognition-Guidance*, using our system prototype and its previously described powerful simulation interfaces, extending and combining our simulation system prototype with DA methods hence seems to be promising future research and development work. Still being a field of active research, an interesting open research question to investigate on is whether it is possible to obtain simulation results of equally high quality from (A) our current, relatively simple MV simulation algorithm (or a similar one) in combination with intelligently exploited medical patient data and thereon-working DA techniques, and (B) a sophisticated, advanced, dedicated MV simulation tool [SES⁺15].

Having accomplished such a DA-enhanced patient-individual MV simulation, validation and evaluation of the overall system setup (e.g., with respect to other cardiac simulation research works as listed in Chapter 3, or with respect to real medical patient data) will then make sense, too, and likely yield rewarding results.

As a final remark, we emphasize that from the impact of DA-induced calibration, a system may learn data-intrinsic uncertainties and probability distributions of specific model parameters, which in turn may possibly be represented in a respective modeling and simulation ontology.

Besides the above suggested major items for future work in research and development, we have presented several more ideas and options for further enhancement and improvement with respect to the proposed cognition-guided, simulation-based surgery assistance in the Outlook Sections of the preceding Chapters 4 to 7), as well as in the Open Challenges Section 3.2.

As a final conclusion, in the future, as faster, more sophisticated, more realistic, stable, robust and reliable, patient-specific, cognition-guided surgery simulation techniques are developed and as these become seamlessly embedded and integrated into clinic workflows, with well-accessible user interfaces, FE modeling and simulation has the potential to become the surgeons' best friend and to provide them with advanced knowledge about a possibly best operation for their respective patients.

A Appendix – List of Publications

- Nicolai Schoch, Stefanie Speidel, York Sure-Vetter, Vincent Heuveline
Towards Semantic Simulation for Patient-Specific Surgery Assistance
Online-Proceedings of the International Workshop on Surgical Data Science SDS2016, (access via: <http://www.surgical-data-science.org/>), 2016.
- Nicolai Schoch, Fabian Kießler, Markus Stoll, Sandy Engelhardt, Raffaele De Simone, Ivo Wolf, Rolf Bendl, Vincent Heuveline
Comprehensive patient-specific information preprocessing for cardiac surgery simulations
International Journal of Computer Assisted Radiology and Surgery (IJCARS), Special Issue: IPCAI2016, 11(6):1051–1059, (DOI: [10.1007/s11548-016-1397-0](https://doi.org/10.1007/s11548-016-1397-0)), 2016.
- Nicolai Schoch, Patrick Philipp, Tobias Weller, Sandy Engelhardt, Mykola Volovyk, Andreas Fetzer, Marco Nolden, Raffaele De Simone, Ivo Wolf, Maria Maleshkova, Achim Rettinger, Rudi Studer, and Vincent Heuveline
Cognitive tools pipeline for assistance of mitral valve surgery
Proceedings of SPIE. Medical Imaging 2016: Image-Guided Procedures, Robotic Interventions and Modeling, (DOI: [10.1117/12.2216059](https://doi.org/10.1117/12.2216059)), 2016.
- Andreas Fetzer, Jasmin Metzger, Darko Katic, Keno Maerz, Martin Wagner, Patrick Philipp, Sandy Engelhardt, Tobias Weller, Sascha Zelzer, Alfred Michael Franz, Nicolai Schoch, Vincent Heuveline, Maria Maleshkova, Achim Rettinger, Stefanie Speidel, Ivo Wolf, Hannes Kenngott, Arianeb Mehrabi, Beat Müller-Stich, Lena Maier-Hein, Hans-Peter Meinzer, and Marco Nolden
Towards an open-source semantic data infrastructure for integrating clinical and scientific data in cognition-guided surgery
Proceedings of SPIE. Medical Imaging 2016: PACS and Imaging Informatics: Next Generation and Innovations, (DOI: [10.1117/12.2217163](https://doi.org/10.1117/12.2217163)), 2016.
- Nicolai Schoch, Fabian Kießler, Markus Stoll, Sandy Engelhardt, Raffaele De Simone, Ivo Wolf, Rolf Bendl, Vincent Heuveline
Comprehensive Pre- and Post-Processing for Numerical Simulations in Cardiac Surgery Assistance
Preprint Issue (No. 02/2015), in the Preprint Series of the Engineering Mathematics and Computing Lab (EMCL), (DOI: [10.11588/emclpp.2015.02.23121](https://doi.org/10.11588/emclpp.2015.02.23121)), 2015.
- Nicolai Schoch, Sandy Engelhardt, Raffaele De Simone, Ivo Wolf, Vincent Heuveline
High Performance Computing for Cognition-Guided Cardiac Surgery: Soft Tissue Simulation for Mitral Valve Reconstruction in Knowledge-based Surgery Assistance (Accepted.)

- Modeling, Simulation and Optimization of Complex Processes, Proceedings of the 6th International Conference on High-Performance Scientific Computing HPSC2015, (DOI: not yet known, in press, by Springer-Verlag), 2015.
- Jonas Kratzke, Nicolai Schoch, Christian Weis, Matthias Müller-Eschner, Stefanie Speidel, Mina Farag, Carsten Beller, and Vincent Heuveline
Enhancing 4D PC-MRI in an aortic phantom considering numerical simulations
 Proceedings of SPIE. Medical Imaging 2015: Physics of Medical Imaging, (DOI: [10.1117/12.2082483](https://doi.org/10.1117/12.2082483)), 2015.
 - Nicolai Schoch, Sandy Engelhardt, Norbert Zimmermann, Stefanie Speidel, Raffaele De Simone, Ivo Wolf, and Vincent Heuveline
Integration of a biomechanical simulation for mitral valve reconstruction into a knowledge-based surgery assistance system
 Proceedings of SPIE. Medical Imaging 2015: Image-Guided Procedures, Robotic Interventions and Modeling, (DOI: [10.1117/12.2082153](https://doi.org/10.1117/12.2082153)), 2015.
 - Christoph Paulus, Stefan Suwelack, Nicolai Schoch, Stefanie Speidel, Rüdiger Dillmann and Vincent Heuveline
Simulation of Complex Cuts in Soft Tissue with the Extended Finite Element Method (X-FEM)
 Preprint Issue (No. 02/2014), in the Preprint Series of the Engineering Mathematics and Computing Lab (EMCL), (DOI: [10.11588/emclpp.2014.02.17635](https://doi.org/10.11588/emclpp.2014.02.17635)), 2014.
 - Nicolai Schoch, Fabian Kißler
HiFlow³-Tutorial: Elasticity Simulation using HiFlow³
 HiFlow³-Tutorial, (access via: <http://hiflow3.org/index.php?id=37>), 2014.
 - Nicolai Schoch, Sandy Engelhardt, Norbert Zimmermann, Stefanie Speidel, Raffaele De Simone, Ivo Wolf, and Vincent Heuveline
Integration einer Biomechanischen Simulation zur Unterstützung von Mitralklappen-Rekonstruktionen in ein Wissensbasiertes Assistenzsystem
 Article in Tagungsband der CURAC 2014, Deutsche Gesellschaft für Computer- und Roboterassistierte Chirurgie, e.V., 2014.
 - Stefan Suwelack, Markus Stoll, Sebastian Schalck, Nicolai Schoch, Rüdiger Dillmann, Rolf Bendl, Vincent Heuveline, and Stefanie Speidel
The Medical Simulation Markup Language (MSML) – Simplifying the Biomechanical Modeling Workflow
 Proceedings of the 21st Medicine Meets Virtual Reality MMVR2014, (DOI: [10.3233/978-1-61499-375-9-394](https://doi.org/10.3233/978-1-61499-375-9-394)), 2014.
 - Nicolai Schoch, Stefan Suwelack, Rüdiger Dillmann and Vincent Heuveline
Simulation of Surgical Cutting of Soft Tissue using the X-FEM
 Preprint Issue (No. 04/2013), in the Preprint Series of the Engineering Mathematics and Computing Lab (EMCL), (DOI: [10.11588/emclpp.2013.04.11825](https://doi.org/10.11588/emclpp.2013.04.11825)), 2013.

Bibliography

- [ARK06] A. Abou-Rjeili and G. Karypis. Multilevel algorithms for partitioning power-law graphs. *20th International Parallel and Distributed Processing Symposium 2006 (IPDPS)*, 2006.
- [ABB⁺15] B.M. Adams, L.E. Bauman, W.J. Bohnhoff, K.R. Dalbey, M.S. Ebeida, J.P. Eddy, M.S. Eldred, P.D. Hough, K.T. Hu, J.D. Jakeman, J.A. Stephens, L.P. Swiler, D.M. Vigil, and T.M. Wildey. Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.0 user’s manual. *Sandia Technical Report SAND2014-4633*, 2015.
- [AAR⁺06] D.H. Adams, A.C. Anyanwu, P.B. Rahmanian, V. Abascal, S.P. Salzberg, and F. Filsoufi. Large annuloplasty rings facilitate mitral valve repair in Barlow’s Disease. *Ann Thorac Surg*, 82(6):2096–2101, 2006.
- [AAB⁺10] H. Anzt, W. Augustin, M. Baumann, H. Bockelmann, T. Gengenbach, T. Hahn, V. Heuveline, E. Ketelaer, D. Lukarski, A. Otzen, S. Ritterbusch, B. Rucker, S. Ronnas, M. Schick, C. Subramanian, J.-P. Weiss, and F. Wilhelm. HiFlow³ – a flexible and hardware-aware parallel finite element package. *EMCL Preprint Series*, (06), 2010.
- [AAB⁺12] H. Anzt, W. Augustin, M. Baumann, T. Gengenbach, T. Hahn, A. Helfrich-Schkarbanenko, V. Heuveline, E. Ketelaer, D. Lukarski, A. Nestler, S. Ritterbusch, S. Ronnas, M. Schick, M. Schmidtobreck, C. Subramanian, J.-P. Weiss, F. Wilhelm, and M. Wlotzka. HiFlow³ – a hardware-aware parallel finite element package. *Tools for High Performance Computing 2011*, pages 139–151, 2012.
- [AHR10] H. Anzt, T. Hahn, V. Heuveline, and B. Rucker. GPU accelerated scientific computing: Evaluation of the NVIDIA Fermi architecture; elementary kernels and linear solvers. *EMCL Preprint Series*, (04), 2010.
- [ASS15] R. Arp, B. Smith, and A.D. Spear. *Building ontologies with Basic Formal Ontology*. MIT Press, 2015.
- [Ate16] G.A. Ateshian. *Mixture theory for modeling biological tissues: Illustrations from articular cartilage*, pages 1–51. Springer, 2016.
- [BTP⁺06] J. Bacon, N. Tardella, J. Pratt, J. Hu, and J. English. The Surgical Simulation and Training Markup Language (SSTML): an xml-based language for medical simulation. *Stud Health Technol Inform*, 119:37–42, 2006.

- [BBE⁺08] S. Balay, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, and H. Zhang. PETSc users manual. *Technical Report ANL-95/11 - Revision 3.0.0*, 2008.
- [Bal77] J.M. Ball. Convexity conditions and existence theorems in non-linear elasticity. *Archive for Rational Mechanics and Analysis*, 63:337–403, 1977.
- [Bal02] J.M. Ball. *Some open problems in elasticity*, In: *Geometry, Mechanics, and Dynamics*, pages 3–59. Springer New York, 2002.
- [BNSH06] D. Balzania, P. Neff, J. Schröder, and G.A. Holzapfel. A polyconvex framework for soft biological tissues. adjustment to experimental data. *International Journal of Solids and Structures*, 43(20):6052–6070, 2006.
- [BHH⁺16] W. Bangerth, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, and B. Turcksin. The deal.II library, version 8.3. *Archive of Numerical Software*, 4(100):1–11, 2016.
- [BW98] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1998.
- [BA09] S.D. Barnett and N. Ad. Surgery for aortic and mitral valve disease in the United States: a trend of change in surgical practice between 1998 and 2005. *J Thorac Cardiovasc Surg*, 137(6):1422–1429, 2009.
- [Bat96] K.-J. Bathe. *Finite Element Procedures*. Springer, 1996.
- [BLME13] T. Belytschko, W.K. Liu, B. Moran, and K. Elkhodary. *Nonlinear finite elements for continua and structures*. Wiley, 2013.
- [Ber06] J. Bergmann. Health and clinical management – comparing the clinical outcome of computerized strategies: Findings from the section on health and clinical management. *IMIA Yearbook*, 2006.
- [BL06] T. Berners-Lee. Linked Data. design issues 2006-07-27. retrieved 2010-12-18. *W3C. Design Issues*, 2006.
- [BLHL01] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*, 284:34–43, 2001.
- [BBF⁺15] A. Biem, M. Butrico, M. Feblowitz, T. Klinger, Y. Malitsky, K. Ng, A. Perer, C. Reddy, A. Riabov, H. Samulowitz, D. Sow, G. Tesauero, and D. Turaga. Towards cognitive automation of data science. *AAAI Conference on Artificial Intelligence*, 2015.
- [BHBL09] C. Bizer, T. Heath, and T. Berners-Lee. Linked Data – the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.

- [BW08] J. Bonet and R. Wood. *Nonlinear continuum mechanics for finite element analysis, 2nd edition*. Cambridge University Press, 2008.
- [BMD13] W. Bothe, D.C. Miller, and T. Doenst. Sizing for mitral annuloplasty: Where does science stop and voodoo begin? *Ann Thorac Surg*, 95(4):1475–1483, 2013.
- [Bra85] R. Brachman. *Readings in Knowledge Representation*. In: Ronald Brachman and Hector J. Levesque. *Readings in Knowledge Representation*, by Morgan Kaufmann, 1985.
- [Bra07] D. Braess. *Finite Elemente*. Springer, Berlin Heidelberg, 2007.
- [BN98] M. Bro-Nielsen. Finite element modeling in Medical VR. *J. IEEE Biomechanical Modeling for Image Registration*, 86(3):490–503, 1998.
- [CB97] R. Carey and G. Bell. The annotated VRML 2.0 reference manual. 1997.
- [CAF10] A. Carpentier, D. Adams, and F. Filsoufi. *Carpentier’s Reconstructive Valve Surgery*. Saunders W.B. Saunders/Elsevier, 2010.
- [CP04] M. Chabanas and E. Promayon. Physical Model Language: Towards a unified representation for continuous and discrete models. *Medical Simulation*, pages 256–266, 2004.
- [CP11] V. Chan and A. Perlas. Basics of ultrasound imaging, in: S.n. narouze (ed.), atlas of ultrasound-guided procedures in interventional pain management. pages 13–19, 2011.
- [CK14] K.B. Chandran and H. Kim. Computational mitral valve evaluation and potential clinical applications. *Annals of Biomedical Engineering*, 43(6):1348–1362, 2014.
- [CG07] D.H. Chang and B.F. Goodrich. Rheology and processing of polymeric materials: Volume 1: Polymer rheology. *Rheology and Processing of Polymeric Materials*, 1, 2007.
- [CB86] A.B. Chaudhary and K.-J. Bathe. A solution method for static and dynamic analysis of three-dimensional contact problems with friction. *J. f. Comp. Struct.*, 24:855–873, 1986.
- [CMES10] E.S. Chen, G.B. Melton, M. Engelstad, and I.N. Sarkar. Standardizing clinical document names using the HL7/LOINC document ontology and LOINC codes. *AMIA Annu Symp Proc. 2010*, pages 101–105, 2010.
- [CMMN04] L. Chen, A.D. McCulloch, and K. May-Newman. Nonhomogeneous deformation in the anterior leaflet of the mitral valve. *Ann. Biomed. Eng.*, 32(12):1599–1606, 2004.

- [CYMN04] L. Chen, F.C.P. Yin, and K. May-Newman. The structure and mechanical properties of the mitral valve leaflet-strut chordae transition zone. *J. Biomech. Engng.*, 126:244–251, 2004.
- [CHBC12] R.C. Chen, Y.H. Huang, C.T. Bau, and S.M. Chen. A recommendation system based on domain ontology and SWRL for anti-diabetic drugs selection. *Expert Systems with Applications*, 29(4):3995–4006, 2012.
- [CLL98] S. Chen, K. Liu, and Z. Liu. Spectrum and stability for elastic systems with global or local Kelvin-Voigt Damping. *SIAM Journal on Applied Mathematics*, 59(2):651–668, 1998.
- [CPWS09] K.H. Cheung, E. Prud’hommeaux, Y. Wang, and S. Stephens. Semantic Web for health care and life sciences: a review of the state of the art. *Briefings in Bioinformatics*, 10(2):111–113, 2009.
- [CRMK14] A. Choi, Y. Rim, J.S. Mun, and H. Kim. A novel finite element-based patient-specific mitral valve repair: virtual ring annuloplasty. *Journal of Bio-Medical Materials and Engineering*, 24:341–347, 2014.
- [Cia88] P.G. Ciarlet. *Mathematical Elasticity - Volume I: Three-Dimensional Elasticity*. North-Holland, 1988.
- [CK92] A.O. Cifuentes and A. Kalbag. A performance study of tetrahedral and hexahedral elements in 3-D finite element structural analysis. *Finite Elements in Analysis and Design*, 12(3):313–318, 1992.
- [CCC⁺08] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. Meshlab: an open-source mesh processing tool. *Proc. Eurographics Italian Chapter Conference*, pages 129–136, 2008.
- [CA88] A. Curnier and P. Alart. A generalized Newton Method for contact problems with friction. *J. of Mec. Theor. Appl.*, 7:67–82, 1988.
- [DKNO10] A. Dedner, R. Kloforn, M. Nolte, and M. Ohlberger. A generic interface for parallel and adaptive scientific computing: Abstraction principles and the DUNE-FEM module. *Computing*, 90(4):165–196, 2010.
- [DLVFM⁺10] A. Diaz-Lantada, R.D. Valle-Fernandez, P.L. Morgado, J. Munoz-Garcia, J.L. Munoz Sanz, J.M. Munoz-Guijosa, and J.E. Otero. Development of personalized annuloplasty rings: combination of CT images and CAD-CAM tools. *Ann Biomed Eng*, 38(2):280–290, 2010.
- [Dub13] Open Organization DublinCore. Dublin core metadata initiative. 2013.
- [EGDM15] H. Ebrahimi, Y. Gao, H. Dozono, and K. Muramatsu. Comparison of time integration methods in magnetomechanical problems. *IEEE Transactions on Magnetics*, 51(3):1–4, 2015.

- [EdSAM⁺16] S. Engelhardt, R. de Simone, S. Al-Maisary, S. Kolb, M. Karck, H.-P. Meinzer, and I. Wolf. Accuracy evaluation of a mitral valve surgery assistance system based on optical tracking. *International Journal of Computer Assisted Radiology and Surgery*, 11(10):1891–1904, 2016.
- [ELAM⁺15] S. Engelhardt, N. Lichtenberg, S. Al-Maisary, R. de Simone, H. Rauch, J. Roggenbach, S. Mueller, M. Karck, H.-P. Meinzer, and I. Wolf. Towards automatic assessment of the mitral valve coaptation zone from 4d ultrasound. *Proc. Functional Imaging and Modelling of the Heart (FIMH) 2015*, 2015.
- [EMSH97] C.R. Ethier, J.A. Moore, D.A. Steinman, and D. Holdsworth. Construction of vascular models from MRI: accuracy assessment based on flow simulations in a carotid bifurcation model. *Int. J. Cardiovasc. Med. Sci*, 1(1):73 ff, 1997.
- [Eva10] L.C. Evans. *Partial Differential Equations*. American Mathematical Society, 2010.
- [FP09] A. Fabri and S. Pion. CGAL: The Computational Geometry Algorithms Library. *Proc. 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 538–539, 2009.
- [FP16] A. Faldum and K. Pommerening. An optimal code for patient identifiers, in: *Computer methods and programs in biomedicine*. 79(1):81–88, 2016.
- [FJY06] R.D. Falgout, J.E. Jones, and U.M. Yang. The design and implementation of Hypre, a library of parallel high performance preconditioners. *Lecture Notes in Computational Science and Engineering*, 51:267–294, 2006.
- [FDD⁺12] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, and I. Peterlik. SOFA: A multi-model framework for interactive physical simulation. *Proc. Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, pages 283–321, 2012.
- [FMB08] P.W.M. Fedak, P.M. McCarthy, and R.O. Bonow. Evolving concepts and technologies in mitral valve repair. *Circulation*, 117(7):963–974, 2008.
- [FL15] J. Feinberg and H.P. Langtangen. ChaosPy: An open source tool for designing methods of uncertainty quantification. *Journal of Computational Science*, 11:46–57, 2015.
- [FFD⁺11] T. Feldman, E. Foster, Glower D.D., S. Kar, M.J. Rinaldi, P.S. Fail, R.W. Smalling, R. Siegel, G.A. Rose, E. Engeron, C. Loghin, A. Trento, E.R. Skipper, T. Fudge, G.V. Letsou, J.M. Massaro, and L. Mauri. Percutaneous repair or surgery for mitral regurgitation. *New England Journal of Medicine*, 364(15):1395–1406, 2011.
- [FMK⁺16] A. Fetzer, J. Metzger, D. Katic, K. März, M. Wagner, P. Philipp, S. Engelhardt, T. Weller, S. Zelzer, A.M. Franz, N. Schoch, V. Heuveline, M. Maleshkova,

- A. Rettinger, S. Speidel, I. Wolf, H. Kenngott, A. Mehrabi, B. Müller, L. Maier-Hein, H.-P. Meinzer, and M. Nolden. Towards an open-source semantic data infrastructure for integrating clinical and scientific data in cognition-guided surgery. *Proc. SPIE. Medical Imaging 2016*, 9789:978900–978908, 2016.
- [FHB03] W. Flameng, P. Herijgers, and K. Bogaerts. Recurrence of mitral valve regurgitation after mitral valve repair in degenerative valve disease. *Circulation*, 107(12):1609–1613, 2003.
- [Fun93] Y.C. Fung. *Biomechanics: mechanical properties of living tissues*. Springer, 1993.
- [GSG⁺09] J.S. Gammie, S. Sheng, B.P. Griffith, E.D. Peterson, J.S. Rankin, S.M. O’Brien, and J.M. Brown. Trends in mitral valve surgery in the United States: results from the society of thoracic surgeons adult cardiac surgery database. *Ann. Thorac. Surg.*, 87(5):1431–1437, 2009.
- [GMQ⁺14] H. Gao, X. Ma, N. Qi, C. Berry, B.E. Griffith, and X. Luo. A finite strain nonlinear human mitral valve model with fluid structure interaction. *Int. J. Numer. Meth. Biomed. Engng*, 30:1597–1613, 2014.
- [GSH14] S. Gawlok, C. Straub, and V. Heuveline. Time-discretization methods based on convection-diffusion equation. *HiFlow³ Tutorials*, 2014.
- [GMW⁺14] L. Ge, W.G. Morrel, A. Ward, R. Mishra, Z. Zhang, J.M. Guccione, and M.B. Ratcliffe. Measurement of mitral leaflet and annular geometry and stress after repair of posterior leaflet prolapse: Virtual repair using a patient specific finite element simulation. *Annals of Thoracic Surgery*, 97(5):1496–1503, 2014.
- [GMP⁺14] P. Gemmeke, M. Maleshkova, P. Philipp, M. Götz, C. Weber, B. Kämpgen, M. Nolden, K.H. Maier-Hein, and A. Rettinger. Using Linked Data and Web APIs for automating the pre-processing of medical images. *Proc. ISWC Workshop on Consuming Linked Data (COLD)*, 2014.
- [GW08] J. Georgii and R. Westermann. Corotated finite elements made fast and stable, in: Proc. 5th ws virtual reality interaction and physical simulation. 2008.
- [GR15] M. Geradin and D. Rixen. *Mechanical Vibrations: Theory and Applications to Structural Dynamics, Third Edition*. John Wiley and Sons, 2015.
- [GNK⁺01] D.T. Gering, A. Nabavi, R. Kikinis, N. Hata, L.J. O’Donnell, W.E.L. Grimson, P.M. Black F.A. Jolesz, and W.M. Wells. An integrated visualization system for surgical planning and guidance using image fusion and an open mr. *Journal of Magnetic Resonance Imaging*, 13(6):967–975, 2001.
- [Gon85] T. Gonzalez. Clustering to minimize the maximum intercluster distance. *Journal of Theoretical Computer Science*, 38:293–306, 1985.

- [GSAM⁺13] B. Graser, M. Seitel, S. Al-Maisary, M. Grossgasteiger, T. Heye, H.-P. Meinzer, D. Wald, R. de Simone, and I. Wolf. *Computer-Assisted Analysis of Annuloplasty Rings*, In: *BVM 2013*, pages 75–80. Springer Berlin Heidelberg, 2013.
- [GWAM⁺14] B. Graser, D. Wald, S. Al-Maisary, M. Grossgasteiger, R. de Simone, H.-P. Meinzer, and I. Wolf. Using a shape prior for robust modeling of the mitral annulus on 4d ultrasound data. *IJCARS*, 9(4):635–644, 2014.
- [GIV⁺12] S. Grbic, R. Ionasec, D. Vitanovski, I. Voigt, Y. Wang, B. Georgescu, N. Navab, and D. Comaniciu. Complete valvular heart apparatus model from 4D Cardiac CT. *Medical Image Analysis*, 16(5):1003–1014, 2012.
- [GMSM13] H. Greenspan, H. Müller, and T. Syeda-Mahmood. Putting Watson to Work: Watson in Healthcare. *IBM*. Retrieved November 11, 2013, 2013.
- [Gri12] B.E. Griffith. Immersed boundary model of aortic heart valve dynamics with physiological driving and loading conditions. *International Journal for Numerical Methods in Biomedical Engineering*, 28(3):317–345, 2012.
- [GHW03] G. Guizzardi, H. Herre, and G. Wagner. On the general ontological foundations of conceptual modeling. In: *Conceptual Modeling – ER 2002*, 2003.
- [GW10] G. Guizzardi and G. Wagner. Using the Unified Foundational Ontology (UFO) as a foundation for general conceptual modeling languages. *Theory and Applications of Ontology: Computer Applications*, pages 175–196, 2010.
- [HSdNH12] P.E. Hammer, M.S. Sacks, P.J. del Nido, and R.D. Howe. Mass-spring model for simulation of heart valve tissue mechanical behavior. *Annals of Biomedical Engineering*, 39(6):1668–1679, 2012.
- [HN03] S. Hartmann and P. Neff. Polyconvexity of generalized polynomial type hyperelastic strain energy functions for near incompressibility. *International Journal of Solids and Structures*, 40(11):2767–2791, 2003.
- [HGS03] M. Hauth, J. Gross, and W. Strasser. Interactive physically based solid dynamics. *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation*, 2003.
- [HS04] M. Hauth and W. Strasser. Corotational simulation of deformable solids. *Journal of WSCG*, 12(1-3), 2004.
- [HPS14] P.J. Hayes and P.F. Patel-Schneider. Resource description framework (RDF) 1.1 semantics. *W3C Recommendation, published at 25 February 2014*, 2014.
- [HRWL83] F. Hayes-Roth, D.A. Waterman, and D.B. Lenat. *Building Expert Systems*. Addison-Wesley Longman Publishing Co., Inc., 1983.
- [HM09] T. Heimann and H.-P. Meinzer. Statistical shape models for 3d medical image segmentation: A review. *Med Image Anal*, 13(4):543–563, 2009.
- [HAL04] A. Henderson, J. Ahrens, and C. Law. *The ParaView guide*. 2004.

- [HS90] N.J. Higham and R.S. Schreiber. Fast polar decomposition of an arbitrary matrix. *SIAM J. Sci. Stat. Comput.*, 11(4):648–655, 1990.
- [HHT77] H.M. Hilber, T.J.R. Hughes, and R.L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering and Structural Dynamics*, 5(3):283–292, 1977.
- [HKP⁺12] P. Hitzler, M. Kroetzsch, B. Parsia, P.F. Patel-Schneider, and S. Rudolph. OWL 2 Web Ontology Language: Primer (second edition). *W3C Recommendation, published at 12 December 2012*, 2012.
- [Hol00] G.A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Wiley, England, 2000.
- [HGO00] G.A. Holzapfel, T.C. Gasser, and R.W. Ogden. A new constitutive framework for arterial wall mechanics and a comparative study of material models. *J. Elasticity*, 68:1–48, 2000.
- [HO16] G.A. Holzapfel and R.W. Ogden, editors. *Biomechanics – Trends in Modelling and Simulation*. Springer, 2016.
- [Iai09] P.A. Iaizzo, editor. *The Handbook of Cardiac Anatomy, Physiology, and Devices. 2nd edition*. Humana Press Inc., New York, NY, 2009.
- [Ibr09] A. Ibrahimbegovic. *Nonlinear solid mechanics: theoretical formulations and finite element solution methods*. Springer, 2009.
- [IVG⁺10] R. Ionasec, I. Voigt, B. Georgescu, Y. Wang, H. Houle, F. Vega Higuera, N. Navab, and D. Comaniciu. Patient-specific modeling and quantification of the aortic and mitral valves from 4D Cardiac CT and TEE. *IEEE Trans Med Imaging*, 1636–1651:29(9), 2010.
- [IVG⁺09] R. Ionasec, I. Voigt, B. Georgescu, Y. Wang, H. Houle, J. Hornegger, Navab N., and D. Comaniciu. Personalized modeling and assessment of the aortic-mitral coupling from 4D TEE and CT. *Proc. Med Image Comput Comput Assist Interv (MICCAI) 2009, Part II*, pages 767–775, 2009.
- [IUBH10] R. Isele, J. Umbrich, C. Bizer, and A. Harth. LDspider: An open-source crawling framework for the web of Linked Data. *9th International Semantic Web Conference (ISWC2010)*, 2010.
- [JIM⁺11] A. Jabbour, T.F. Ismail, N. Moat, A. Gulati, I. Roussin, F. Alpendurada, B. Park, F. Okoroafor, A. Asgar, S. Barker, S. Davies, S.K. Prasad, M. Rubens, and R.H. Mohiaddin. Multimodality imaging in transcatheter aortic valve implantation and post-procedural aortic regurgitation: comparison among cardiovascular magnetic resonance, cardiac computed tomography, and echocardiography. *Journal of the American College of Cardiology*, 58(21):2165–2173, 2011.

- [Jäh05] Bernd Jähne. *Digital Image Processing. 6th revised and extended edition*. Springer, 2005.
- [JBV⁺11] A.S. Jassar, C.J. Brinster, M. Vergnat, J.D. Robb, T.J. Eperjesi, A.M. Pouch, A.T. Cheung, S.J. Weiss, M.A. Acker, and J.H. III Gorman. Quantitative mitral valve modeling using real-time three-dimensional echocardiography: Technique and repeatability. *The Annals of Thoracic Surgery*, 91(1):165–171, 2011.
- [JHPH07] J.J. Jokinen, M.J. Hippelaeinen, O.A. Pitkaenen, and J.E. Hartikainen. Mitral valve replacement versus repair: Propensity-adjusted survival and quality-of-life analysis. *Journal of Thoracic Surgery*, 84(2):451–458, 2007.
- [KMV⁺13] J. Kanik, T. Mansi, I. Voigt, P. Sharma, R.I. Ionasec, D. Comaniciu, and J. Duncan. Estimation of patient-specific material properties of the mitral valve using 4d transesophageal echocardiography. *2013 IEEE 10th International Symposium on Biomedical Imaging*, pages 1178–1181, 2013.
- [KK98] G. Karypis and V. Kumar. A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. *J. Parallel Distrib. Comput.*, 48(1):71–95, 1998.
- [KJW⁺15] D. Katić, C. Julliard, A.-L. Wekerle, H. Kenngott, B. Müller-Stich, R. Dillmann, S. Speidel, P. Jannin, and B. Gibaud. LapOntoSPM: an ontology for laparoscopic surgeries and its application to surgical phase recognition. *International Journal of Computer Assisted Radiology and Surgery*, 10(9):1427–1434, 2015.
- [KO88] N. Kikkuchi and J.T. Oden. Contact problems in elasticity: A study of variational inequalities and finite element methods. *SIAM, Philadelphia*, 1988.
- [Kla88] A. Klarbring. On discrete and discretized nonlinear elastic structures in unilateral contact (stability, uniqueness and variational principles). *International Journal of Solids and Structures*, 24:459–479, 1988.
- [KIS⁺09] G. Krishnamurthy, A. Itoh, J.C. Swanson, W. Bothe, M. Karlsson, E. Kuhl, D. Craig Miller, and N.B. Jr. Ingels. Regional stiffening of the mitral valve anterior leaflet in the beating ovine heart. *Journal of Biomechanics*, 42(16):2697–2701, 2009.
- [KVV06] M. Krötzsch, D. Vrandečić, and M. Völkel. *Semantic MediaWiki, In: The Semantic Web - ISWC 2006*, pages 935–942. Springer Berlin Heidelberg, 2006.
- [KCC⁺93] K.S. Kunzelman, R.P. Cochran, C. Chuong, W.S. Ring, E.D. Verrier, and R.D. Eberhart. Finite element analysis of the mitral valve. *J Heart Valve Dis*, 2(3):326–340, 1993.
- [KCVE94] K.S. Kunzelman, R.P. Cochran, E.D. Verrier, and R.C. Eberhart. Anatomic basis for mitral valve modelling. *J Heart Valve Dis*, 3(5):491–496, 1994.
- [LTH⁺10] P. Lancellotti, C. Tribouilloy, A. Hagendorff, L. Moura, B. Popescu, E. Agricola, J. Monin, L. Pierard, L. Badano, J. Zamorano, R. Sicari, A. Vahanian, and

- J. Roelandt. European association of echocardiography recommendations for the assessment of valvular regurgitation. part 1: Aortic and pulmonary regurgitation (native valve disease). *Eur J Echocardiogr*, 11(3):223–244, 2010.
- [LDSB10] K.D. Lau, V. Diaz, P. Scambler, and G. Burriesci. Mitral valve dynamics in structural and fluid-structure interaction models. *Medical Engineering and Physics*, 32(9):1057–1064, 2010.
- [LS13] T.B. Le and F. Sotiropoulos. Fluid-structure interaction of an aortic heart valve prosthesis driven by an animated anatomic left ventricle. *Journal of Computational Physics*, 244:41–62, 2013.
- [LMK10] O.P. Le Maitre and O.M. Knio. *Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics*. Springer, 2010.
- [LAG⁺13] C.H. Lee, R. Amini, R.C. Gorman, J.H. Gorman, and M.S. Sacks. An inverse modeling approach for stress estimation in mitral valve anterior leaflet valvuloplasty for in-vivo valvular biomaterial assessment. *Journal of Biomechanics*, 47(9):2055–2063, 2013.
- [Lue84] D.G. Luenberger. Linear and nonlinear programming. *Addison-Wesley, Reading, Mass., second edition*, 1984.
- [MEAW12] S.A. Maas, B.J. Ellis, G.A. Ateshian, and J.A. Weiss. FEBio: finite elements for biomechanics. *Journal of biomechanical engineering*, 134(1), 2012.
- [MACP02] A. Magkanaraki, S. Alexaki, V. Christophides, and D. Plexousaki. Benchmarking RDF Schemas for the Semantic Web. *In Proc. ISWC 2002*, 2002.
- [MHMK⁺14] L. Maier-Hein, S. Mersmann, D. Kondermann, S. Bodenstedt, A. Sanchez, C. Stock, H. Kenngott, M. Eisenmann, and S. Speidel. Can masses of non-experts train highly accurate image classifiers? a crowdsourcing approach to instrument segmentation in laparoscopic images. *Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI) 2014, Part II*, pages 349–356, 2014.
- [MRS⁺05] F. Maisano, A. Redaelli, M. Soncini, E. Votta, L. Arcobass, and O. Alfieri. An annular prosthesis for the treatment of functional mitral regurgitation: finite element model analysis of a dog bone-shaped ring prosthesis. *Ann Thorac Surg*, 79(4):1268–1275, 2005.
- [MSD⁺09] F. Maisano, R. Skantharaja, P. Denti, A. Giacomini, and O. Alfieri. Mitral annuloplasty. *Oxford Journals Multimedia Manual of Cardio-Thoracic Surgery*, 0918, 2009.
- [MVG⁺12] T. Mansi, I. Voigt, B. Georgescu, X. Zheng, E.A. Mengue, M. Hackl, R.I. Ionasec, T. Noack, J. Seeburger, and D. Comaniciu. An integrated framework for finite element modeling of mitral valve biomechanics from medical images. *Journal of Medical Image Analysis*, 16(7):1330–1346, 2012.

- [MORB07] D.S. Marcus, T.R. Olsen, M. Ramaratnam, and R.L. Buckner. The extensible neuroimaging archive toolkit: an informatics platform for managing, exploring, and sharing neuroimaging data. *Neuroinformatics*, 5(1):11–34, 2007.
- [Mar14] G. Marom. Numerical methods for Fluid-Structure Interaction models of aortic valves. *Archives of Computational Methods in Engineering*, 22(4):1–26, 2014.
- [MH83] J.E. Marsden and J.R. Hughes. *Mathematical Foundations of Elasticity*. Prentice Hall, 1983.
- [MO87] J.A.C. Martins and J.T. Oden. Existence and uniqueness results for dynamic contact problems with nonlinear normal and friction interface laws. *Nonlinear Analysis, Theory, Methods and Applications*, 11:407–428, 1987.
- [MHW⁺15] K. März, M. Hafezi, T. Weller, A. Saffari, M. Nolden, N. Fard, A. Majlesara, S. Zelzer, Maleshkova M., M. Volovyk, N. Gharabaghi, M. Wagner, G. Emami, S. Engelhardt, A. Fetzner, H. Kenngott, N. Rezai, A. Rettinger, A. Studer, R. Mehrabi, and L. Maier-Hein. Toward knowledge-based liver surgery: holistic information processing for surgical decision support. *Int J Comput Assist Radiol Surg*, 10(6):749–759, 2015.
- [MKT⁺15] S. Mate, F. Koepcke, D. Toddenroth, M. Martin, H.-U. Prokosch, T. Buerkle, and T. Ganslandt. Ontology-based data integration between clinical and research systems. *PLoS ONE*, 10(1), 2015.
- [MSW⁺15] S. Mazaheri, P.S. Sulaiman, R. Wirza, M.Z. Dimon, F. Khalid, and R.M. Tayebi. Uncertainty estimation for improving accuracy of non-rigid registration in cardiac images. *Proc. 11th Artificial Intelligence Applications and Innovations Conference*, pages 19–28, 2015.
- [MGB⁺04] E.C. McGee, A.M. Gillinov, E.H. Blackstone, J. Rajeswaran, G. Cohen, F. Najam, T. Shiota, J.F. Sabik, B.W. Lytle, P.M. McCarthy, and D.M. Cosgrove. Recurrent mitral regurgitation after annuloplasty for functional ischemic mitral regurgitation. *J. Thorac. Cardiovasc. Surg.*, 128(6):916–924, 2004.
- [Mei15] A. Meister. *Numerik linearer Gleichungssysteme. Eine Einführung in moderne Verfahren*. Springer, 2015.
- [Mez08] J. Mezger. Simulation and animation of deformable bodies. *Dissertation thesis at University of Tübingen, Germany*, 2008.
- [MTPS08] J. Mezger, B. Thomaszewski, S. Pabst, and W. Strasser. Interactive physically-based shape editing. *ACM Solid and Physical Modeling Symposium (SPM)*, pages 79–89, 2008.
- [Mil56] G.A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97, 1956.

- [MHC07] J.A. Miller, C. He, and J.I Couto. Impact of the Semantic Web on modeling and simulation. *Handbook of Semantic System Modeling*, 2007.
- [Moo40] M. Mooney. A theory of large elastic deformation. *Journal of Applied Physics*, 11(9):582–592, 1940.
- [MPW⁺16] A.E. Morgan, J.L. Pantoja, J. Weinsaft, E. Grossi, J.M. Guccione, L. Ge, and M. Ratcliffe. Finite element modeling of mitral valve repair. *J Biomech Eng*, 138(2):(in press), 2016.
- [MGZ⁺14] W.G. Morrel, L. Ge, Z. Zhang, E.A. Grossi, J.M. Guccione, and M.B. Ratcliffe. Effect of mitral annuloplasty device shape and size on leaflet and myofiber stress following repair of posterior leaflet prolapse: A patient-specific finite element simulation. *J. Heart Valve Dis*, 23(6):727–734, 2014.
- [MHG⁺03] R.R. Moss, K.H. Humphries, M. Gao, C.R. Thompson, J.G. Abel, G. Fradet, and B.I. Munt. Outcome of mitral valve repair or replacement: a comparison by propensity score analysis. *Circulation*, 108 Suppl1:I:90–97, 2003.
- [MMD⁺02] M. Müller, L. McMillan, J. Dorsey, R. Jagnow, and B. Cutler. Stable real-time deformations. *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation*, pages 49–54, 2002.
- [Mur15] J.W. Murdock. Decision making in IBM Watson Question Answering. *Ontology Summit 2015. Web presentations*, 2015.
- [MWM⁺10] S.N. Murphy, G. Weber, M. Mendis, V. Gainer, H.C. Chueh, S. Churchill, and I. Kohane. Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). *Journal of the American Medical Informatics Association*, 17(2):124–130, 2010.
- [NOB⁺14] R.A. Nishimura, C.M. Otto, R.O. Bonow, B.A. Carabello, J.P. Erwin, R.A. Guyton, P.T. O’Gara, C.E. Ruiz, N.J. Skubas, P. Sorajja, T.M. Sundt, and J.D. Thomas. 2014 aha/acc guideline for the management of patients with valvular heart diseasea report of the american college of cardiology/american heart association task force on practice guidelines. *Journal of the American College of Cardiology*, 63(22):e57–e185, 2014.
- [NGT⁺06] V.T. Nkomo, J.M. Gardin, Skelton T.N., J.S. Gottdiener, C.G. Scott, and M. Enriquez-Sarano. Burden of valvular heart diseases: a population-based study. *Lancet*, 368(9540):1005–1011, 2006.
- [NZS⁺13] M. Nolden, S. Zelzer, A. Seitel, D. Wald, M. Muller, A.M. Franz, D. Maleike, M. Fangerau, M. Baumhauer, L. Maier-Hein, K.H. Maier-Hein, H.-P. Meinzer, and I. Wolf. The Medical Imaging Interaction Toolkit: challenges and advances: 10 years of open-source development. *Int J CARS*, 8:607–620, 2013.

- [NMMH15] T. Norajitra, H.-P. Meinzer, and K.H. Maier-Hein. 3d statistical shape models incorporating 3d random forest regression voting for robust CT liver segmentation. *Proc. SPIE. Medical Imaging 2015*, 9414:941406–1–941406–6, 2015.
- [OGS09] D. Oberle, N. Guarino, and S. Staab. What is an ontology? *In: Handbook on Ontologies, 2nd edition*, 2009.
- [Ogd97] R.W. Ogden. *Non-linear Elastic Deformations*. Courier Corporation, 1997.
- [PTJP14] M. Panahiazar, V. Taslimitehrani, A. Jadhav, and J. Pathak. Empowering personalized medicine with Big Data and Semantic Web technology: Promises, challenges, and use cases. *Proc. IEEE International Conference on Big Data 2014*, 10:790–795, 2014.
- [PO92] D. Peric and D.R.J. Owen. Computational model for 3-D contact problems with friction based on the penalty method. *J. for Numerical Methods in Engineering*, 35:1289–1309, 1992.
- [PK15] I. Peterlik and A. Klima. Towards an efficient data assimilation in physically-based medical simulations. *In: Proc. IEEE International Conference on Bioinformatics and Biomedicine (BTBM)*, 2015.
- [PMK⁺15] P. Philipp, M. Maleshkova, D. Katic, C. Weber, M. Götz, A. Rettinger, S. Speidel, B. Kämpgen, M. Nolden, A.-L. Wekerle, R. Dillmann, H. Kenngott, B. Müller, and R. Studer. Toward cognitive pipelines of medical assistance algorithms. *Int J CARS*, November 2015.
- [PHK04] S. Pieper, M. Halle, and R. Kikinis. 3D Slicer. *Proc. IEEE International Symposium on Biomedical Imaging: Nano to Macro 2004*, pages 632–635, 2004.
- [PR04] K. Pommerening and M. Reng. Secondary use of the EHR via pseudonymisation. *Studies in Health Technology and Informatics*, 103:441–6, 2004.
- [PXY⁺12] A. Pouch, C. Xu, P.A. Yushkevich, A.S. Jassar, M. Vergnet, J.H. III Gorman, C.M. Sehgal, and B.M. Jackson. Semi-automated mitral valve morphometry and computational stress analysis using 3d ultrasound. *Journal of Biomechanics*, 45(9):903–907, 2012.
- [PHS09] V. Prot, R. Haaverstad, and B. Skallerud. Finite element analysis of the mitral apparatus: annulus shape effect and chordal force distribution. *Biomech. Model. Mechanobiol.*, 8:43–55, 2009.
- [PS09] V. Prot and B. Skallerud. Nonlinear solid finite element analysis of mitral valves with heterogeneous leaflet layers. *J. of Comput. Mech.*, 43:353–368, 2009.
- [PSSH10] V. Prot, B. Skallerud, G. Sommer, and G.A. Holzapfel. On modelling and analysis of healthy and pathological human mitral valves: two case studies. *Journal of the Mechanical Behavior of Biomedical Materials*, 3:167–177, 2010.

- [QV08] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations (1st edition)*. Springer, 2008.
- [RMOc86] P.J. Rabier, J.A.C. Martins, J.T. Oden, and L. Campos. Existence and local uniqueness of solutions to contact problems in elasticity with nonlinear friction laws. *International Journal of Engineering Science*, 24:1755–1768, 1986.
- [Ran08] R. Rannacher. Numerik 3. *Lecture Notes at the University of Heidelberg*, 2008.
- [Ran13] R. Rannacher. Numerik 1. *Lecture Notes at the University of Heidelberg*, 2013.
- [Rec08] A. Rector. Barriers, approaches and research priorities for integrating biomedical ontologies. *Semantic Health Deliverable 6.1*, 2008.
- [Rob11] E.H. Robinson. A theory of social agentivity and its integration into the descriptive ontology for linguistic and cognitive engineering. *International Journal on Semantic Web and Information Systems*, 7(4):62–86, 2011.
- [RM07] C. Rosse and J.L.V. Mejino. The Foundational Model of Anatomy Ontology. In: A. Burger, D. Davidson, and R. Baldock, editors, *Anatomy Ontologies for Bioinformatics: Principles and Practice*, 6:59–117, 2007.
- [RN10] S.J. Russell and P. Norvig. Artificial Intelligence: A modern approach. *Upper Saddle River, New Jersey: Prentice Hall*, pages 437–439, 2010.
- [RCB⁺07] A. Ruttenberg, T. Clark, W. Bug, M. Samwald, O. Bodenreider, H. Chen, D. Doherty, K. Forsberg, Y. Gao, V. Kashyap, J. Kinoshita, J. Luciano, M.S. Marshall, C. Ogbuji, J. Rees, S. Stephens, G.T. Wong, E. Wu, D. Zaccagnini, T. Hongsermeier, E. Neumann, I. Herman, and K.-H. Cheung. Advancing translational research with the Semantic Web. *BMC Bioinformatics*, 8(3):1–16, 2007.
- [Saa03] Y. Saad. *Iterative methods for sparse linear systems. 2nd edition*. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, 2003.
- [SHLM14] M. Schick, V. Heuveline, and O.P. Le Maitre. A newton-galerkin method for fluid flow exhibiting uncertain periodic dynamics. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):153–173, 2014.
- [SRHE15] J. Schindelin, C.T. Rueden, M.C. Hiner, and K.W. Eliceiri. The ImageJ ecosystem: An open platform for biomedical image analysis. *Molecular Reproduction and Development*, 82(7-8):518–529, 2015.
- [SPV⁺12] R.J. Schneider, D.P. Perrin, N.V. Vasilyev, G.R. Marx, P.J. del Nido, and R.D. Howe. Mitral annulus segmentation from 4d ultrasound using a valve state predictor and constrained optical flow. *Medical Image Analysis*, 16(2):497–504, 2012.
- [SES⁺15] N. Schoch, S. Engelhardt, R. De Simone, I. Wolf, and V. Heuveline. High performance computing for cognition-guided cardiac surgery: Soft tissue simulation

- for mitral valve reconstruction in knowledge-based surgery assistance. *Modeling, Simulation and Optimization of Complex Processes, In: Proc. High Performance Scientific Computing (HPSC) 2015*, 2015.
- [SEZ⁺15] N. Schoch, S. Engelhardt, N. Zimmermann, S. Speidel, R. de Simone, I. Wolf, and V. Heuveline. Integration of a biomechanical simulation for mitral valve reconstruction into a knowledge-based surgery assistance system. *Proc. SPIE. Medical Imaging 2015*, 9415:941502–941502–7, 2015.
- [SKH15] N. Schoch, F. Kißler, and V. Heuveline. Elasticity tutorial for soft tissue simulation. *HiFlow³ Tutorials*, 2015.
- [SKS⁺15] N. Schoch, F. Kißler, M. Stoll, S. Engelhardt, R. de Simone, I. Wolf, R. Bendl, and V. Heuveline. Comprehensive pre- and post-processing for numerical simulations in cardiac surgery assistance. *EMCL Preprint Series*, (02), 2015.
- [SKS⁺16] N. Schoch, F. Kißler, M. Stoll, S. Engelhardt, R. de Simone, I. Wolf, R. Bendl, and V. Heuveline. Comprehensive patient-specific information preprocessing for cardiac surgery simulations. *Int J CARS*, 11(6):1051–1059, 2016.
- [SPW⁺16] N. Schoch, P. Philipp, T. Weller, S. Engelhardt, M. Volovyk, A. Fetzer, M. Nolden, R. de Simone, I. Wolf, M. Maleshkova, A. Rettinger, R. Studer, and V. Heuveline. Cognitive tools pipeline for assistance of mitral valve surgery. *Proc. SPIE. Medical Imaging 2016*, 9786:978603–978603–8, 2016.
- [SSSVH16] N. Schoch, S. Speidel, Y. Sure-Vetter, and V. Heuveline. Towards semantic simulation for patient-specific surgery assistance. *In: Online-Proceedings of Surgical Data Science 2016*, 2016.
- [SSS⁺13] N. Schoch, S. Suwelack, S. Speidel, R. Dillmann, and V. Heuveline. Simulation of surgical cutting of soft tissue using the X-FEM. *EMCL Preprint Series*, (04), 2013.
- [SCW⁺11] A. Scott, W. Courtney, D. Wood, R. De la Garza, S. Lane, R. Wang, M. King, J. Roberts, J.A. Turner, and V.D. Calhoun. COINS: An innovative informatics and neuroimaging tool suite built for large heterogeneous datasets. *Frontiers in Neuroinformatics*, 5(33), 2011.
- [SGP⁺13] Suendermann. S.H., M. Gessat, N. Perrin, G.R. Marx, T. Frauenfelder, P. Biaggi, D. Bettex, F. Volkmar, and S. Jacobs. Implantation of personalized, biocompatible mitral annuloplasty rings: feasibility study in an animal model. *Interact Cardiovasc Thorac Surg*, 16(4):417–422, 2013.
- [Si15] H. Si. TetGen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw.*, 41(2):11:1–11:36, 2015.
- [SHM07] G.A. Silver, O.A.-H. Hassan, and J.A. Miller. From domain ontologies to modeling ontologies to executable simulation models. *In: Proc. WSC2007 Winter Simulation Conference 2007*, pages 1108–1117, 2007.

- [SH98] J.C. Simo and J.R. Hughes. *Computational Inelasticity*. Springer, 1998.
- [SL92] J.C. Simo and T.A. Laursen. An augmented Lagrangian treatment of contact problems involving friction. *Computers and Structures*, 42:97–116, 1992.
- [SSS⁺15] S. Skornitzke, G. Schummers, M. Schreckenber, J. Ender, S. Eibel, H.-J. Bungartz, H.-U. Kauczor, and W. Stiller. Mass-spring systems for simulating mitral valve repair using 3d ultrasound images. *Comp. Med. Imag. and Graph.*, 45:26–35, 2015.
- [Sot12] F. Sotiropoulos. Computational fluid dynamics for medical device design and evaluation: Are we there yet? *Cardiovascular Engineering and Technology*, 3(2):137–138, 2012.
- [SSHS13] S. Stadtmüller, S. Speiser, A. Harth, and R. Studer. Data-Fu: A language and an interpreter for interaction with read/write Linked Data. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1225–1236. International World Wide Web Conferences Steering Committee, 2013.
- [SKV⁺11] M. Stevanella, G. Krishnamurthy, E. Votta, J.C. Swanson, A. Redaelli, and N.B. Ingels. Mitral leaflet modeling: Importance of in vivo shape and material properties. *J Biomech*, 44(12):2229–2235, 2011.
- [SMC⁺11] M. Stevanella, F. Maffessanti, C.A. Conti, E. Votta, A. Arnoldi, M. Lombardi, O. Parodi, E.G. Caiani, and A. Redaelli. Mitral valve patient-specific finite element modeling from Cardiac MRI: Application to an annuloplasty procedure. *Cardiovascular Engineering and Technology*, 2(2):66–76, 2011.
- [SOV⁺14] F. Sturla, F. Onorati, E. Votta, K. Pechlivanidis, M. Stevanella, A.D. Milano, G. Puppini, A. Mazzucco, A. Redaelli, and G. Faggian. Is it possible to assess the best mitral valve repair in the individual patient? preliminary results of a finite element study from magnetic resonance imaging data. *J. Thorac. Cardiovasc. Surg.*, 148(3):1025–1034, 2014.
- [SVO⁺15] F. Sturla, E. Votta, F. Onorati, K. Pechlivanidis, O.A. Pappalardo, L. Gottin, A.D. Milano, G.Puppini, A. Redaelli, and G. Faggian. Biomechanical drawbacks of different techniques of mitral neochordal implantation: When an apparently optimal repair can fail. *J. Thorac. Cardiovasc. Surg.*, 150(5):1303–1312, 2015.
- [SM03] E. Süli and D.F. Mayers. *An introduction to numerical analysis*. Cambridge University Press, 2003.
- [Sul15] T.J. Sullivan. *Introduction to Uncertainty Quantification*. Springer, 2015.
- [Suw14] S. Suwelack. *Real-time Biomechanical Modeling for Intraoperative Soft Tissue Registration (Dissertation)*. KIT Scientific Publishing, 2014.
- [SSS⁺14] S. Suwelack, M. Stoll, S. Schalck, N. Schoch, R. Dillmann, R. Berndl, V. Heuveline, and S. Speidel. The medical simulation markup language – simplifying the

- biomechanical modeling workflow. *Journal on Studies in Health Technology and Informatics*, 196:394–400, 2014.
- [SKI⁺11] J.C. Swanson, G. Krishnamurthy, A. Itoh, J.P. Kvitting, W. Bothe, D. Craig Miller, and N.B. Jr. Ingels. Multiple mitral leaflet contractile systems in the beating heart. *Journal of Biomechanics*, 44(7):1328–1333, 2011.
- [SVIH16] S. Swaroop-Vedula, M. Ishii, and G.D. Hager. Perspectives on Surgical Data Science. In: *Online-Proceedings of Surgical Data Science 2016*, 2016.
- [TRDB10] L. Temal, A. Rosier, O. Dameron, and A. Burgun. Mapping BFO and DOLCE. *Studies in health technology and informatics*, 160(2):1065–1069, 2010.
- [THF⁺13] N.A. Tenenholtz, P.E. Hammer, A. Fabozzo, E.N. Feins, P.J. del Nido, and R.D. Howe. Fast simulation of mitral annuloplasty for surgical planning. *Proc. Functional Imaging and Modeling of the Heart (FIMH) 2013*, 7945:106–113, 2013.
- [TKT⁺13] C. Ting, R. Kumar, G.A. Troianowski, T. Syeda-Mahmood, D. Beymer, and K. Brannon. PSAR: Predictive Space Aggregated Regression and its application in valvular heart disease classification. *IEEE International Symposium on Biomedical Imaging*, pages 1122–1125, 2013.
- [TSF07] N. Toussaint, J.-C. Souplet, and P. Fillard. MedINRIA: Medical image navigation and research tool by INRIA. *Proc. MICCAI 2007*, 7, 2007.
- [TN04] C. Truesdell and W. Noll. *The Non-Linear Field Theories of Mechanics*. Springer, 2004.
- [TT60] C. Truesdell and R. Toupin. *The classical field theories*. Springer, 1960.
- [Tur04] Z.G. Turi. Cardiology patient page: Mitral valve disease. *Circulation*, 109(6):38–41, 2004.
- [TL94] G. Turk and M. Levoy. Zippered polygon meshes from range images. *Proc. Annual Conference Series, ACM SIGGRAPH 1994*, pages 311–318, 1994.
- [VH01] R. Vdovjak and G.-J. Houben. RDF-based architecture for semantic integration of heterogeneous information sources. In *In Workshop on Information Integration on the Web*, pages 51–57, 2001.
- [Ver13] R. Verfürth. *A posteriori error estimation techniques for finite element methods*. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, 2013.
- [V.G13] Ramanathan V.G. Light at the End of the Tunnel (ISWC 2013 keynote talk). In *Proc. ISWC 2013*, 2013.
- [VIG⁺09] I. Voigt, R.I. Ionasec, B. Georgescu, H. Houle, M. Huber, J. Hornegger, and D. Comaniciu. Model-driven physiological assessment of the mitral valve from 4D TEE. *Proc. SPIE Medical Imaging 2009*, 7261, 2009.

- [VMI⁺11] I. Voigt, T. Mansi, R. Ionasec, E. A. Mengue, H. Houle, B. Georgescu, J. Hornegger, and D. Comaniciu. Robust physically-constrained modeling of the mitral valve and subvalvular apparatus. *Proc. MICCAI 2011*, 2011.
- [VCV⁺08] E. Votta, E. Caiani, F. Veronesi, M. Soncini, F.M. Montevecchi, and A. Redaelli. Mitral valve finite-element modelling from ultrasound data: a pilot study for a new approach to understand mitral function and clinical scenarios. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 366(1879):3411–3434, 2008.
- [VLS⁺13] E. Votta, T.B. Le, M. Stevanella, L. Fusini, E.G. Caiani, A. Redaelli, and F. Sotiropoulos. Toward patient-specific simulations of cardiac valves: State-of-the-art and future directions. *Journal of Biomechanics*, 46:217–228, 2013.
- [VMB⁺07] E. Votta, F. Maisano, S.F. Bolling, O. Alfieri, F.M. Montevecchi, and A. Redaelli. The Geoform disease-specific annuloplasty system: a finite element study. *Ann. Thorac. Surg.*, 84(1):92–101, 2007.
- [WS12] Q. Wang and W. Sun. Finite element modeling of mitral valve dynamic deformation using patient-specific multi-slices computed tomography scans. *Annals of Biomedical Engineering*, 41(1):141–153, 2012.
- [WZC⁺10] J.F. Wenk, Z. Zhang, G. Cheng, D. Malhotra, G. Acevedo-Bolton, M. Burger, T. Suzuki, D.A. Saloner, A.W. Wallace, J.M. Guccione, and M.B. Ratcliffe. First finite element model of the left ventricle with mitral valve: insights into ischemic mitral regurgitation. *Annals of Thoracic Surgery*, 89(5):1546–1553, 2010.
- [WK03] B. Wohlmuth and R. Krause. Monotone multigrid methods on nonmatching grids for nonlinear multibody contact problems. *SIAM Journal on Scientific Computing*, 25(1):324–347, 2003.
- [WWZ⁺12] V.M. Wong, J.F. Wenk, Z. Zhang, G. Cheng, G. Acevedo-Bolton, M. Burger, D.A. Saloner, A.W. Wallace, J.M. Guccione, M.B. Ratcliffe, and L. Ge. The effect of mitral annuloplasty shape in ischemic mitral regurgitation: A finite element simulation. *Ann. Thorac. Surg.*, 93(3):776–782, 2012.
- [Wri06] P. Wriggers. Computational contact mechanics. 2nd edition. *Springer, Berlin Heidelberg*, 2006.
- [WM92] P. Wriggers and C. Miehe. On the treatment of contact constraints within coupled thermomechanical analysis. *Proc. of EUROMECH, Finite Inelastic Deformations*, 1992.
- [XPP00] C. Xu, D.L. Pham, and J.L. Prince. *Handbook of Medical Imaging – Volume 2: Medical Image Processing and Analysis*. SPIE Press, 2000.
- [ZT00] O.C. Zienkiewicz and R.L. Taylor. The finite element method. *Oxford: Butterworth-Heinemann*, 1-3, 2000.