# Realtime Cardiac Surgery Simulation

Jesper Mosegaard, mosegard@daimi.au.dk

10th March 2004

# Abstract

This thesis is an investigation of the possibilities of realtime surgical simulation, specifically within the domain of congenital heart diseases. The problem domain of surgery on the heart of an infant is presented and the primary information necessary for a pre-operative surgical simulation tool is identified. Through cooperation with surgeons, domain specific abstractions and simplifications with respect to soft tissue simulation is identified and used in the investigation of models for tissue deformation.

In general it is not possible to calculate the deformation of tissue precisely in realtime. We therefore need to use simplified elastic models that approximate the real behavior of tissue. The most popular models for realtime interaction and response; the Finite Element model and the Spring Mass model are presented and discussed with respect to to surgery on the heart of an infant. A number of variations and extensions of the classic Spring Mass model are investigated to identify the characteristics that can use the domain specific knowledge of tissue deformation and interaction in cardiac surgery. An elastic model and a geometrical model build specifically for cardiac surgery simulation is developed and presented in this thesis.

The thesis is done in cooperation with surgeons from Århus University hospital. A surgical simulator supporting the techniques discussed in the thesis has been evaluated with surgeons for pre-operative use. Furthermore the elastic-models under investigation have been compared and validated formally with respect to their behavior of deformation.

# Acknowledgement

First of all thanks to the staff at the department of Cardiothoracic and Vascular Surgery at Aarhus University Hospital for allowing me to observe the surgical procedures and explaining the details. A special thanks to Ole Kromann for his dedication and initiatives in relation to the work presented in this thesis.

Thanks to Peter Møller Nielsen for guidance through the thesis and his dedication to the subject. A special thanks to Thomas Sangild Sørensen for his inspiration, initiatives and helpful discussions.

Thanks to Thomas Sangild Sørensen, Rory Andrew Wright Middleton and Steffen Brandorff for proof readings.

Thanks to the Oticon Foundation for the scholarship I received, enabling me to work full time on this thesis and participating in the Medicine Meets Virtual Reality 2003 conference in Los Angeles and ED-MEDIA 2003 conference in Honolulu.

# Contents


**1 Introduction** **1**

**I Problem Domain** **3**

**2 Congenital Cardiac Defects** **4**
2.1 Interdisciplinary fields . . . . . 4
2.2 The learning process . . . . . 5
2.3 Surgical Procedures on the heart . . . . . 5
2.4 The heart . . . . . 5
2.5 Surgical procedures . . . . . 6
2.5.1 Opening the chest . . . . . 7
2.5.2 Ventricular Septal Defect (VSD) . . . . . 7
2.5.3 Atrium Septum Defect (ASD) . . . . . 7
2.6 Observations . . . . . 8
2.6.1 Small forces and small deformations . . . . . 8
2.6.2 Areas of interest . . . . . 8
2.6.3 Controlled movements . . . . . 8
2.6.4 Cutting procedure . . . . . 9
2.7 Goals and cases . . . . . 9
2.7.1 Three generations of surgical tools . . . . . 9
2.7.2 Goals as defined by the Surgeon . . . . . 10
2.8 Categories of usage . . . . . 10
2.8.1 Pre-operative planning . . . . . 11
2.8.2 Education Scenarios and training. . . . . 11
2.8.3 Skill assessment . . . . . 12
2.9 Contributions . . . . . 12

**3 The Research Field** **13**
3.1 Datasets and Segmentation . . . . . 13
3.2 Representation . . . . . 14
3.3 Soft Tissue Modeling . . . . . 15
3.4 Interaction and Haptics . . . . . 16
3.5 Visualization and Display . . . . . 17

3.5.1 Usage and Validation . . . . . 17

**II Surgical Simulation 18**

**4 Finite Element Models 19**
4.1 Theory of Elasticity . . . . . 20
4.2 The Energy Function . . . . . 21
4.3 Discretization of the Energy Function using Finite Elements . . . 23
4.3.1 Creating the global stiffness matrix K . . . . . 24
4.4 Finding the minimum energy configuration . . . . . 25
4.5 Solving the linear system of equations . . . . . 25
4.5.1 Fast solving . . . . . 26
4.5.2 Supporting realtime cuts . . . . . 27

**5 Spring Mass Models 30**
5.1 Spring mass formulation . . . . . 30
5.2 Internal forces . . . . . 32
5.2.1 Home forces . . . . . 32
5.2.2 Nonlinear force models . . . . . 33
5.2.3 Volume preservation . . . . . 33
5.3 Solving the second order differential equation . . . . . 33
5.3.1 Explicit Euler Integration . . . . . 34
5.3.2 Runga Kutta . . . . . 34
5.3.3 Verlet . . . . . 34
5.3.4 Stability . . . . . 35
5.4 Static equilibrium . . . . . 35
5.5 Point interaction . . . . . 36
5.6 Relaxation . . . . . 38
5.7 Local interaction in large models . . . . . 39
5.8 Hardware speed-up . . . . . 40
5.9 Spring topology issues . . . . . 41
5.9.1 No best way . . . . . 42
5.9.2 Surface models . . . . . 43
5.9.3 True volumetric model . . . . . 44
5.9.4 Connected Surfaces . . . . . 44
5.10 Evaluation of Spring Mass in comparison with FEM . . . . . 46
5.11 3D Chainmail . . . . . 47

**6 Design Issues of a Unified Soft Tissue Simulation 50**
6.1 ElasticObject . . . . . 50
6.2 Geometries . . . . . 53

**7 Interface: Interaction and Visualization 55**
7.1 Tools . . . 55
7.2 Collision detection . . . 56
7.3 Functionality of instruments . . . 56
7.4 Abstract tools . . . 57
7.5 Visualization . . . 59

**8 Altering Topology of Tissue 60**
8.1 Strategies for cutting . . . 61
8.2 Removal of elements . . . 61
8.3 Subdividing tetrahedrons . . . 62
8.4 Splitting along faces in the elements . . . 63
8.4.1 Selecting faces to unglue . . . 63
8.4.2 Ungluing faces . . . 64
8.4.3 Node snapping . . . 65
8.4.4 Degeneracy removal . . . 65
8.5 Cutting in Connected Surfaces . . . 65

**III Validation 68**

**9 Parameter Optimization 69**
9.1 EA for parameter optimization . . . 70
9.1.1 Comparing models . . . 71
9.1.2 Chromosomes . . . 74
9.2 Minimum and Maximum fitness . . . 75
9.3 Tetrahedral QuasiStatic to QuasiStatic optimization . . . 75
9.4 Tetrahedral QuasiStatic to precise FEM optimization . . . 77
9.5 Tetrahedral Relaxation . . . 78
9.6 Tetrahedral QuasiStatic with Relaxation . . . 79
9.7 Dynamic Spring Mass . . . 81
9.8 Quasti Static Connected Surfaces . . . 82
9.9 Comparison of models . . . 83

**10 Evaluation with Surgeons 87**
10.1 Deformation equilibrium . . . 87
10.2 Dynamic or static model . . . 88
10.3 Deformation animation . . . 88
10.4 Interaction . . . 89
10.5 Simplifications . . . 89
10.6 Categories of Use . . . 90

**11 Conclusion 91**

**12 Future Research** **92**
12.1 Further evaluation with surgeons . . . . . 92
12.2 Validation . . . . . 92
12.3 Experimental comparison of models . . . . . 93
12.4 Seeded iterative models . . . . . 93
12.5 Specialized physical models . . . . . 94

**A FEM Details** **101**
A.1 $K^e$ is symmetric . . . . . 101
A.2 $B^e$ matrix . . . . . 101

**B Surgical Simulator Implementation** **102**
B.1 CD-ROM . . . . . 102
B.2 Compilation . . . . . 102
B.3 Init files . . . . . 103
B.4 Interaction . . . . . 103
B.5 View . . . . . 103
B.6 Implementation details . . . . . 104

**C Validation and Reliability** **106**
C.1 Validity . . . . . 106
C.2 Reliability . . . . . 107
C.3 Taxonomy . . . . . 107

**D Evaluation from Surgeons** **108**

# List of Figures


2.1 Basic anatomy of the heart (from [13]) . . . . . 6
2.2 VSD (from [13]) . . . . . 7
2.3 ASD (from [13]) . . . . . 8

3.1 The outer surface in transparent with the inner in solid . . . . . 14

4.1 The linear relationship between stress and strain approximating a non-linear relationship. . . . . 21
4.2 Discretization of the shape $\Omega$ into triangle elements e1 to e9. . . 23

5.1 Spring response . . . . . 32
5.2 Local interaction and deformation at the two red points. . . . . 37
5.3 Relaxation on a deformed triangle . . . . . 39
5.4 Dividing deformation into three areas . . . . . 40
5.5 2d regular grid with additional springs . . . . . 41
5.6 Spring to maintain curvature . . . . . 43
5.7 Surface models of heart . . . . . 44
5.8 Forces between surfaces . . . . . 45
5.9 ConnectedSurface for the Heart . . . . . 46
5.10 Flip . . . . . 47
5.11 Deformation of chainmail (from [23]) . . . . . 48

6.1 UML diagram of general design . . . . . 51
6.2 UML diagram of elastic models . . . . . 52
6.3 UML diagram of Geometries . . . . . 53

7.1 UML diagram of Interaction . . . . . 58

8.1 Models for cutting . . . . . 61
8.2 Splitting along faces; selecting faces to unglue, snap nodes to cut-sweep and remove degeneracy. . . . . 63
8.3 Feature selection of a point, an edge and a triangle respectively. . 64
8.4 Cutting in heart . . . . . 67

9.1 The test geometry: A thin wall. (884 nodes, 2408 tetrahedrons and 9766 edges) . . . . . 71
9.2 Nodal fitness mapped onto geometry. Deformation as in figure 9.3. 72
9.3 Test Interaction . . . . . 73
9.4 Fitness landscape of optimization of QuastiStatic parameters to a QuasiStatic model. . . . . 76
9.5 Convergence of the EA for an optimization of QuasiStatic parameters to QuasiStatic model. . . . . 76
9.6 Fitness landscape of optimization of parameters for a QuasiStatic model to FEM (40x40 grid) . . . . . 77
9.7 Convergence of the EA for an optimization of parameters for a QuasiStatic model to a FEM . . . . . 78
9.8 Fitness landscape for an optimization of parameters for a Tetrahedral QuasiStatic with Relaxation to a FEM . . . . . 80
9.9 Convergence of an EA optimizing parameters for a tetrahedral QuasiStatic model with relaxation to a FEM . . . . . 81
9.10 Fitness landscape of the optimization of parameters for a dynamic Spring Mass model in comparison to a FEM . . . . . 82
9.11 Convergence of an EA optimizing parameters for a tetrahedral dynamic Spring Mass model to a FEM . . . . . 83
9.12 Convergence of an EA optimizing parameters for a Connected-Surfaces QuasiStatic Spring Mass model to a FEM . . . . . 84
9.13 Convergence of the elastic models with optimal parameters in comparison to the behavior of a FEM. . . . . 85

# List of Tables


9.1 Minimum stiffness of Tetrahedral QuasiStatic to QuasiStatic optimization. . . . . 75
9.2 Minimum fitness of optimization of parameters for a QuasiStaic model to a FEM . . . . . 77
9.3 Minimum fitness for a parameter optimization for a tetrahedral relaxation (in the range 0 to 3) model compared to a FEM . . . 79
9.4 Minimum fitness for a parameter optimization for a tetrahedral QuasiStatic model with relaxation (in the range 0 to 15) compared to a FEM . . . . . 79
9.5 Minimum fitness for a parameter optimization for a tetrahedral QuasiStatic model with relaxation (in the range 0 to 3) compared to a FEM . . . . . 81
9.6 Minimum fitness for an optimization of parameters for a dynamic Spring Mass model in comparison to a FEM. . . . . 82
9.7 Minimum fitness for a ConnectedSurface Spring Mass model in comparison to a FEM. Stiffness on surface and in connecting springs are the same. . . . . 82
9.8 Minimum fitness for a ConnectedSurface Spring Mass model in comparison to a FEM . . . . . 83
9.9 Framerate of elastic models with the wall model . . . . . 86

# Chapter 1

# Introduction

A simulation is an artificial model of a real procedure, phenomenon or system. The simulation defines rules of behavior that represent the real phenomena to certain degree. Often a simulation is used in training, recreation of real situations or prediction of real world phenomena. There are many reasons to simulate and not actually execute a certain procedure. Generally the real procedure might not be viable economically or ethically, and the elements needed for the procedure might not be easily available.

When simulating a real phenomena we often restrict what parts of the real phenomena we model through the perspective of use. This is especially true because we have limited computational power and must focus on the important parts of the simulation.

A wide variety of simulations have been constructed for different needs. Physical phenomena, such as colliding galaxies and aerodynamic properties are classic examples, but also social phenomena, such as panic in crowds has been simulated to design better emergency plans [30]. In the nineties surgical simulation began to gain respect in the field of surgery [29].

Simulation is perhaps best known from flight simulators. Such simulators are used for education of pilots. The simulation is used instead of flying a real airplane to ensure the safety of people. Furthermore there is a cost associated with bringing an airplane into the air - not to mention the expenses to crash one. In the first flight simulators the simulation focused on the parts that were most important to simulate; the instruments. The flight instruments were connected through some logic that would approximate their behavior in the air. Later more elaborate simulations have been created, but always with a focus on those parts of the real phenomena of flying that are important to the control of an airplane

This thesis will deal with realtime simulation of surgical procedures, from which we would like to learn something about real surgical procedures. More precisely we would like to simulate deformable behavior in the tissues in response to the surgeon interacting with it, either through instruments or with his hands. Specifically we will use the case of congenital heart diseases as the primary

source of problem identification and evaluation. The simulator must support non-destructive interaction such as probing and grasping and also destructive interaction such as cutting and tearing.

The thesis is divided into three parts: Problem Domain, Surgical Simulator and Validation. The Problem Domain includes chapter 2 about the congenital cardiac diseases from which we will derive our cases of use, which are used throughout the thesis. Chapter 3 will present how the different parts of the field of surgical simulation work together and how they are treated in this thesis.

The technical issues of a surgical simulator is discussed and presented in the Surgical Simulator part. Two different tissue models are presented and discussed in chapters 4 and 5. A Surgical Simulator supporting a range of different techniques was developed as part of this thesis and is presented in chapter 6. Chapter 7 presents the general interaction and visualization problems of a surgical simulator. Specifically we will look at how to support topological changes such as cutting in chapter 8.

The last part of Validation will validate and evaluate the surgical simulator in two ways. In chapter 9 I make a formal comparison between the elastic models used in the surgical simulator, and in chapter 10 the evaluation of the Surgical Simulator by surgeons is presented.

# Part I

# Problem Domain

# Chapter 2

# Congenital Cardiac Defects

## 2.1 Interdisciplinary fields

The field of Surgical Simulation is clearly an interdisciplinary field as recognized in e.g. [17]. The expert on surgery is naturally surgeons. They are consequently an important part of a group working with surgical simulation.

On the technical side computer scientists have the skills to analyze and construct the surgical simulator. The technical side includes discretization of bio-mechanical models that can be analyzed with a range of numerical methods resulting in elastic models that can be calculated on a computer. The technical side also includes comparison and analysis of the different elastic models with respect to specific cases of use. Many different themes are of interest to the technical side; these will be summarized in chapter 3.

For this thesis to have some validity, it was important to cooperate with real surgeons. The surgeons would participate in the definition of problem areas and the evaluation of an implementation of a surgical simulator. As part of this thesis I cooperated with pediatric cardiac surgeons Ole Kromann Hansen and Vibeke Hjortdal from the department of Cardiothoracic and Vascular Surgery at Aarhus University Hospital. Kromann and Hjortdal are experienced surgeons in the field of congenital heart disease. The specific surgical procedures used to evaluate the simulator are derived from congenital heart surgery.

An important part of the process of making this thesis has been the interdisciplinary work with the surgeons. In an ideal interdisciplinary work, the participants need to learn about the field of the other participants, to create a common reference. I have therefore learned about and observed the work surgeons do and I have told the surgeons about the technical aspects of a surgical simulator.

Because the heart is our focus, we need to model the geometry of the heart. Thomas Sangild from the MR-Center at Skejby Hospital is currently working with MR-scanning, segmentation and validation in connection to three-dimensional cardiac modeling as part of his PhD project. I cooperated with

Sangild to get the accurate heart morphology as input to the surgical simulation.

## 2.2 The learning process

We will begin with a short introduction to the process of learning surgery. A medical student will have some theoretical knowledge of surgery from the study of medicine. Perhaps he has had the chance to work on a cadaver or a pig. Practicing to become a good surgeon will take many years though, as this is very much a practical skill that has to be learned by doing.

Today, surgery is taught by a master/apprentice principle. The master surgeon will take an apprentice in (often only one) and this apprentice will learn from the master over time. The apprentice will start out observing the master surgeon and will later on be allowed to try basic parts of the procedure. Slowly he will be given more responsibility. In the end, the apprentice is allowed to do entire surgical procedures on his own.

At Århus University Hospital I observed this master/apprentice principle in full. Vibeke Hjortdal is the apprentice of Ole Kromann and started out doing the surgical procedures I saw. She herself had an apprentice, who was allowed to do some basic parts of the surgery and assisting Hjortdal. When problems arose Ole Kromann would be called in to help Vibeke and her apprentice.

## 2.3 Surgical Procedures on the heart

As part of the interdisciplinary work I have observed three surgical procedures at very close range, while the surgeons explained what they where doing. The surgical procedures were afterwards discussed and set in perspective in respect to the simulator. I have made myself familiar with the anatomy of the heart and the surgical procedures with the help of the surgeons and through the online encyclopedia: "The Heart Center Encyclopedia" [13].

The anatomy of the heart and the surgical procedures are presented in the next few sections.

## 2.4 The heart

A simplified drawing of the heart is presented in figure 2.1. The heart has the responsibility of sustaining the circulation of blood in our body, essentially working as a pump. The heart is functionally divided into the right and left part by the septum, each part again divided into an atrium and a ventricle. Between the chambers of the heart are valves to restrict the flow of the blood.

The left side of the heart circulates oxygen-rich blood coming from the lungs to the rest of the body. The pulmonary veins transport the blood from the lungs to the left atrium and the left ventricle transports the blood out to the body via the aorta.

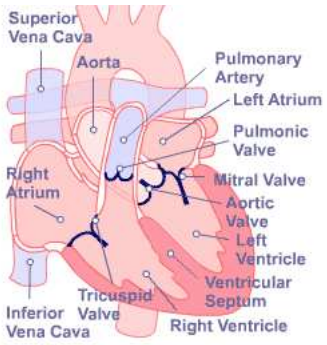



Figure 2.1: Basic anatomy of the heart (from [13])

The right side of the heart circulates oxygen-poor blood from the body to the lungs. The right atrium receives blood from the two largest veins in the body, superior vena cava and inferior vena cava, while the right ventricle sends the blood to the lungs through the pulmonary artery.

Around the heart and the roots of the major blood vessels lies a thin membranes, called the pericardium. Between the heart and the pericardium there is fluid to make the heart move with less resistance due to friction.

As indicated by this very short overview of the heart, the heart is a relatively complex organ, requiring a high degree of geometric detail to represent in a level of detail that resembles reality. Moving one level down, the tissue of the heart is also a complex structure consisting of three layers that have distinct physical characteristics. The tissue is in general non-homogeneous, that is, the physical characteristics is not the same all over the heart, and anisotropic, that is, the deformation of tissue depends on the direction of the force.

## 2.5 Surgical procedures

As part of my cooperation with the surgeons, I observed three surgical procedures dealing with congenital heart defects. Specifically the cases of a Ventricular Septal Defect (VSD) and an Atrial Septal Defect (ASD) have been used as examples in this thesis. Other surgical procedures such as repearing a Patent Ductus Arteriosus (a small vessel that is to be closed) and a Coarctation of the Aorta (too narrow aorta ) were observed.

I observed the surgical procedures from applying anesthesia until the thorax was closed and the skin was stitched together.

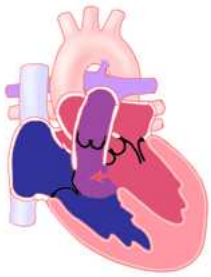


Figure 2.2: VSD (from [13])

### 2.5.1 Opening the chest

The surgical procedures I observed started out in the same way; with a scalpel the surgeon made an incision in the chest to expose the sternum. The sternum[1] was cut open with an electrical saw, and the chest was held open by a clamp throughout the procedure to make a working space for the surgeon.

While the surgeon works on opening the chest, he will inevitable destroy small blood vessels. To minimize bleeding, the surgeon often uses an electrosurgical instrument to make cuts in the tissue and close the blood vessels.

The pericardium is opened next, and stitched to the sternum. This effectively raises the heart from within the chest, creating easier access to the heart.

The next step is to connect the patient to the heart lung machine which will take over the circulation and oxygenation of the blood. Tubes are connected on the major arteries and veins, and the connections from the heart to these vessels are temporarily closed. The heart is stopped, allowing for surgery on the heart.

### 2.5.2 Ventricular Septal Defect (VSD)

This defect is a hole between the right ventricle and the left ventricle of the heart as seen in figure 2.2. In a normal heart the two ventricles are separated. To get to the VSD, the surgeon cuts a hole in the right atrium from which the VSD can be seen. The VSD is typically closed with a patch stitched to the hole.

### 2.5.3 Atrium Septum Defect (ASD)

The ASD, as seen in figure 2.3, is a hole between the right atrium and the left atrium of the heart. Simple ASD's are closed with wires or catheters while others must be closed through surgery. Smaller holes are closed with suturing, while bigger holes require a patch.

[1] bone in the middle of the chest

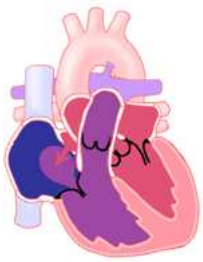


Figure 2.3: ASD (from [13])

## 2.6 Observations

Some basic observations where made and discussed with the surgeons. These observations where used as an initial guide to which models of elasticity and geometry to select.

### 2.6.1 Small forces and small deformations

The heart of an infant or small child is a delicate structure that cannot withhold great stress. The surgeon takes care not to put great stress on the heart. This also means that the deformations of the heart are very small. Because of the material properties of the heart and the very small forces put upon it, the deformations were often local in nature, affecting the shape of the heart in only a relatively small area.

### 2.6.2 Areas of interest

Through conversations and observations in the operating room it was evident that the area in which the surgeon interacts, is well defined and can be rather small, at least for distinct periods of time. Throughout an entire surgical procedure the area of interest will change.

### 2.6.3 Controlled movements

A pattern was recognized in the way the surgeons used the tools in these specific surgergical procedures. The deformations were small and very controlled. The tissue was grabbed and pulled aside either to allow access to other areas, or to the tissue that was grabbed. The access was wanted either for inspection, cutting or sewing.

### 2.6.4 Cutting procedure

The task of cutting is very often perpendicular to the surface. Most sweeps of the scalpel (or other cutting device) are very simple in nature, often short and straight. This behavior comes from the fact that surgeons must be very careful to not cut too much away and generally have to control what is cut and how.

## 2.7 Goals and cases

The primary goal of this thesis is to investigate how a surgical simulator can be used with respect to surgical procedures dealing with congenital heart diseases. Although many of the techniques and solutions discussed can be used in a general surgical simulator, the heart is the ongoing example and reference of surgery. I have identified the special needs that this kind of surgical procedures demands of a simulation and used this perspective as I investigate what work has been done in the field. The surgeon should be able to interact with the simulator in realtime and receive a realtime visualization.

### 2.7.1 Three generations of surgical tools

In order to find out precisely what parts of the surgical procedure we would like to simulate we have to narrow down how much or which aspects of the real phenomena we would need to simulate and represent. One perspective on this in respect to a surgical simulator, is the three generations as presented by Richard Satava [70].

The first generation deals only with geometrical aspects. This generation of surgical tool is not actually a simulation. The concepts introduced and used for learning and pre-operative planning are navigation and immersion in three-dimensional anatomical datasets (as opposite to 2D pictures). In the work by Thomas Sangild it is shown that this kind of use bears great promise [74].

The second generation deals with soft tissue deformation. That is, how tissue deforms in response to some interaction with it.

The third generation deals with the functionality of tissue and organs, e.g. blood flow, electrical signals etc. One example of a third generation simulation is the discussion of the simulation of a beating heart [64]. The idea is to simulate the tissue down to cellular size, computing the electrical signals of a single cell comprising the pulsation of the heart. The electrical signals are computed by a dozen differential equations and there are hundred thousands of coupled cells in a complete heart model. Such a simulation could e.g. be used to test for new drugs that can prevent arrhythmia because the functionality is also simulated. The key point is that knowledge exists that can explain the behavior of tissue exactly. An implementation of a mathematical model of heart mechanics outside the scope of surgical simulation has been presented in [52]. A complex Finite Element model is used as a simulation of a part of the heart beat. Compared to the second generation it is out of scope to make realtime simulations of third generation surgical tools.

I believe that these three generations are not to be thought of as a hierarchy of better or worse tools for learning and information. The three generations represent different perspectives on *what* to learn. As mentioned earlier, one of the strengths of simulation is that we can focus on those parts of reality that are important, and simplify or leave out others parts.

The surgical procedures used as cases in this thesis deal with a heart that has been stopped. I.e. the main functionality of the heart is non-functional because the heart-lung machine has taken over circulation of blood. Furthermore the procedures themselves deal with re-construction of the heart to an improved functionality. The most important part of the simulation should therefore be the soft tissue deformation in response to interactions and cutting - in the perspective of the three generations the thesis deals with the 2. generation.

### 2.7.2 Goals as defined by the Surgeon

The goal for the surgical simulator as defined in this thesis is to simulate tissue response to the tools used in a surgical procedure, specifically a procedure correcting some congenital deformation in a heart. The simulator must run in realtime[2], both with respect to visual feedback and interactivity.

The soft tissue deformation includes the calculation of spatial configurations of the tissue in response to absolute constraints of a number of points in the tissue as well as general external forces affecting the tissue. This means that the interaction of surgical tools used to probe, pinch and stretch can be simulated. The soft tissue deformation must deliver a complete solution for the system to be visualized in realtime on a standard desktop computer.

Another constraint for the system is that it must support such techniques as cutting, burning, ripping etc. That is, altering the topology of the tissue. The surgery involving a deformable heart often has as it's goal to re-model the heart to support the body with blood in a more favorable fashion. This means that the simulation must support altering the original topology besides the elastic behavior due to grasping and probing[3].

The heart has a complex structure and we need a large amount of geometrical detail to represent the shape of the heart.

## 2.8 Categories of usage

Some different categories of use of a general surgical simulator have been discussed with the surgeons.

[2]Realtime visual and interactive respons is often categorized as being above 20 frames per second.

[3]As we shall see later this constraint is severe because it prohibits us from doing precalculations to be used in the soft tissue deformation.

### 2.8.1 Pre-operative planning

The initial need for the surgeons of Skejby Hospital was a tool for them to do pre-operative planning. This means that patient specific heart geometry would be loaded into the simulator and the surgeon could rehearse or collect information from the simulation. The case of rehearsal is very similar to general training and will be discussed in the next section.

A very important point made by the surgeons was that the simulator could be used in the planning of the procedure. When the surgeon analyses information from 2d or even 3d images of the heart one important aspect is missing; the deformation he normally experiences when analyzing the situation in an actual surgical procedure. The surgeon is not used to looking at pure geometrical models - the surgeon looks at models that deform when he is investigating them. He is simply more experienced at making decision based on what he sees in actual surgery, which is an open heart. E.g. when a surgeon looks at an ASD in actual surgery, he decides what to do based on the location of the hole in the atrium septum. He looks at the atrium septum from a hole made in the left atrium, essentially deforming the heart to get a view of the septum. A surgical simulator should ultimately be able to present the surgeon with the same image of the open heart, and based on this he could make the same decision pre-operatively as in the actual surgical situation. A static geometry of a closed heart does not resemble what the surgeon sees.

The simulator can also be used to ensure that a given procedure can be executed as planned. The simulator can give the surgeon information about e.g. the level of stress that the tissue is exposed to or whether a given piece of tissue can cover a hole, or be reconstructed and fitted into a given shape. E.g. in the case of ASD or VSD where the holes are closed with patches.

### 2.8.2 Education Scenarios and training.

A general simulator could be used in training and educational scenarios. In recent years surgical simulation has begun to gain clinical respect and is predicted to be an integrated part of training to become a surgeon [45]. In general educational scenarios, the organ model could be simplified or idealized to support pedagogical points.

The potential of a simulator for training is to minimize risk of patients, standardize the surgery curriculum and train on arbitrary (e.g. rare) anatomies. In section 2.2 learning of surgery procedures was presented. A simulator could support this learning process with the possibility of training surgical procedures. A simulator can relieve the student of time pressure because there is no risk to the patient. A point given by the surgeons was that it is simply difficult to navigate in the i heart, it takes practise to learn it.

The Surgical Simulator can be used as an alternative to surgical training on cadavers or animals. In [37] surgical simulation is presented as an alternative for the "Advanced Trauma Life Support course". Animals do not correctly represent an anatomy for realistic training. Cadavers have the correct anatomy, but are

expensive and can be difficult to acquire. In all cases the cadavers and animals are not reusable and raise ethical issues.

A simulator could be used as a tool in an medical curriculum or as a tool to transfer knowledge. The teachers might use the simulator to present a procedure or technique, and the students could afterwards try out the procedure for themselves.

New or rare surgical procedures can be recorded by experts, thereby sharing their knowledge. Whole libraries of knowledge could be constructed. The student can watch the procedure from any angle and can take over control of the simulator at any time. In this case, a patient specific heart with the rare condition would be used.

The training scenarios must be adequate for the aspects of the surgical procedure that is to be trained. It is not necessarily the ultimate goal to just simulate reality. E.g. in [**?**, 69] a risk reducing training is set up. Risk estimates are used to avoid damage of important tissue. In the training scenario the student can feel the risk areas through haptic feedback. [69] mentions brain and cardiac surgery as examples of procedures with risk areas.

It has been recognized [43] that the next big step in surgical simulation for educational scenarios is a formal verification of the usefulness of training. In [71] Richard Satava presented the progress in the Metrics for Objective Assesment of Surgical Skills Worshop. A Surgical Simulation system must be able to show Validity and Reliability, see appendix C.

### 2.8.3 Skill assessment

Especially in the US skill assesment using surgical simulators has been proposed as a way of grading people or selecting people for a surgical career [71, 35].

## 2.9 Contributions

This thesis contributes to the field of surgical simulation in two areas:

As far as I am aware this is the first surgical simulator dealing with cardiac surgery. This challenge is unique because the morphology is very complex in comparison with other organs. Through combinations of techniques I have made it possible to interact with the heart in realtime. These techniques and several others have been implemented and evaluated with real surgeons for the preoperative planning process.

Secondly I have designed a way of comparing elastic models with respect to their actual behavior over time instead of only their equilibrium. The comparison has been done on a number of elastic models and the results are reported.

# Chapter 3

# The Research Field

This chapter presents the different themes of interest to the technical aspects of a surgical simulator. The general problems and constraints are presented as well as the level to which I will deal with the themes. At the TATRICS 3rd annual presentation Dr. Kevin Montgomery derived some of the common themes in surgery simulation research based on 24 different groups working inside the field [48]. The presentation by Montgomery was used as a guide but the themes have been extended and clarified. The themes are: Datasets, Segmentation, Representation, Simulation Engine, Display, Interaction, Haptics and Usage. This is clearly a technical categorization, as the Usage theme is just a single theme. Usage can be further divided into different aspects of usage such as: Validation, Skill Assessment, Training, Pre-operative information/simulation as described in the previous chapter.

This chapter will serve as an overview of the rest of the thesis which will go into depth with the most interesting aspects of a realtime simulator of pediatric heart surgery.

## 3.1 Datasets and Segmentation

The first thing we need for a surgical simulation is some notion of the shape of the tissue and organ we are about to simulate. To get realistic models we need to base them on real human anatomy. The datasets that describe the tissue can originate from several sources. They can mainly be divided into two categories; patient-specific and general datasets. General datasets is avalible through for example the Visible Human project [1].

One method of creating patient-specific data is to acquire individual morphological information from scanners such as MR-scanners or CT-scanners to retrieve a 3d voxel field. This voxel field needs to be segmented to define blood, muscle and other tissues. The segmentation process often outputs the geometry as a number of surfaces consisting of faces that can be visualized directly. The quality of the data from the segmentation is important for the stability and

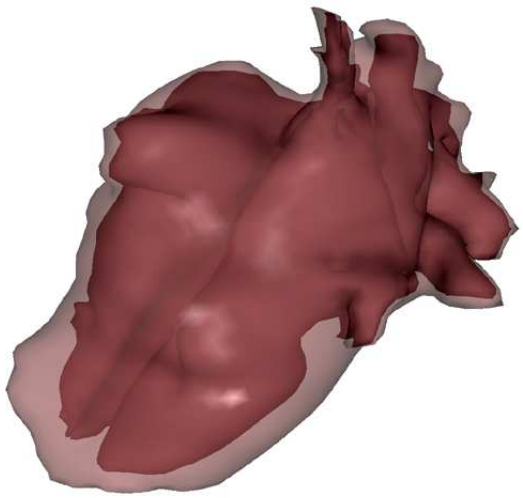


Figure 3.1: The outer surface in transparent with the inner in solid

precision of the simulation.

In this thesis I have used data from the MR-center at Skejby Sygehus and a segmentation program by Søren Vorre and Thomas Sangild [76, 74]. Because of the realtime issue and the quality of the scanning, the heart is approximated as a homogeneous, isotropic heart. Homogeneous meaning that the tissue behaves the same everywhere, and isotropic, that the tissue resists equally to forces in all directions - the heart wall is also a single homogeneous tissue. A lot of manual tuning of the surfaces created by the segmentation program had to be done, because the naive segmentation had some noise and unwanted features. The resulting quality of the segmentation is important because the stability of the soft tissue modeling depends on it. The resulting inner and outer surfaces can be seen in figure 3.1.

## 3.2 Representation

The representation of the tissue is deeply connected to the simulation engine. Essentially it is a conversion of the output from the segmentation to a format that allows the soft tissue simulation to work as fast as possible. The visualization is also a concern in the representation because some structures are more easily visualized than others. Because the heart is a complex structure we need a detailed geometry to represent it to a satisfactory level of detail for it to look like a heart.

Some elastic models demand a tetrahedral mesh. The generation of a tetrahedral mesh from definitions of surfaces is called meshing. In this thesis the

TetGen meshing program has been used to generate tetrahedral meshes from surfaces [31].

The Representation is coupled very tightly to the elastic model chosen. Some models call for volumetric structures consisting of tetrahedrons (e.g. the Finite Element model introduced in chapter 4) , while others can use edges or springs as their basic building tool (e.g. the Spring Mass model, where the topological issues are discussed in section 5.9). The representation used in the actual implementation of the Surgery Simulator is discussed in chapter 6.

## 3.3 Soft Tissue Modeling

Soft Tissue Modeling deals with the issues of bio-mechanical models, and the numerical methods used to calculate the deformations based on the selected models. A tradeoff must be made between *geometrical detail*, *computational speed* and finally *realism and precision*. The Soft Tissue Modeling is the main theme of this thesis.

As mentioned the Soft Tissue modeling is strongly connected to the selected representation, but the soft tissue modeling is also strongly connected to the altering of topology through cuts. Different categories of Simulation Engines are more or less suited for changes in topology.

As explained in the section 3.2, the model needs a lot of geometric detail to represent a heart. The two remaining constraints are computation time and deformation accuracy. The tradeoff between these depends very much on usage as presented in [19]. In a training scenario it is more important to get realtime interaction than absolutely correct results, but in a scientific analysis correct results are of utmost importance. In a planning situation realtime response might be important, but this depends heavily on the specific planning situation. The main focus of this thesis is on realtime aspects. We need to update the surgical simulator with at least 20 frames per second.

> "it doesn't really matter whether the deformation that the surgeon sees in the virtual environment is accurate as long as it seems realistic! Just as important is that the model is robust and shows a consistent and predictable behavior over time" [58]

Inspired by [58] I list the important parts of the realtime soft tissue simulator as:

1. Speed (convergence and update rate)
2. Robustness (consistency, stability and realism)
3. Visual result (realism and graphics)

The speed of the algorithms is essential to the realtime aspect. If the algorithm cannot deliver *some* result within $\frac{1}{20}$ of a second the user will experience too poor a framerate to obtain the illusion of animation.

The second most important aspect is robustness of the algorithm. Robustness covers aspects such as consistency, stability and realism of the deformations displayed. It is intentional that realism is only a part of the robustness demand. If we had unlimited computational power, absolute realism would be equal to robustness - but because computational power is at a shortage and realism can only be approximated, other terms are important too. A simulation should be realistic enough for it to be useful for the categories of usage (see section 2.8). We can relax the degree of realism to a *believable* deformation; it is essential that the range of deformations is consistent (that is that they do not differ much in realism and precision) and perhaps most important that the method is stable. If the numerical methods are not stable there is not much use for it.

In the survey of deformable models [24], a range of different models for the computation of deformable models are presented. In general, very different models exist, both geometrically and physically based. The geometrical models are purely geometric deformations, these deformations are often fast - but have no justification in real physics. This thesis deals with the other category, physically based models. A line can, of course not, be drawn clearly, but we can order the methods as to how well they approximate some physical phenomenon, and to what degree they are meerly geometrical heuristics. Three models are presented in this thesis, the Finite Element, Spring Mass and 3D chainmail. The Finite Element is the most realistic, the Spring Mass is less realistic and the 3D chainmail has very little foundation in physics. The 3D Chainmail is presented as a perspective on the two more physically based models.

Some of the questions I will discuss are the following: Physical realism versus physical plausibility, resolution of the model, accuracy of the deformation dynamics, what deformations can be simulated, interaction, support for topological changes, preprocessing and computational costs. I will also look at the behavior of the models; whether they are dynamic or static. The behavior of the nodes in an elastic model is time dependant, resulting in such effects as waves and vibrations. In a static model there is one equilibrium for a given force and there is as such no notion of mass, damping or inertia. Chapters 4 and 5 deal specifically with two elastic models for soft tissue simulation. The surgical simulator implementation is presented in chapter 6. I will furthermore see how precisely the less realistic models can approximate a more precise model in chapter 9.

## 3.4 Interaction and Haptics

The field of interaction covers the instrumentation of the tissue; e.g. probing, grasping, piercing and suturing. The field of interaction covers both the physical devices, collision detection and response [19] of the soft tissue.

Kinds of interaction that has been given special attention is cutting, tearing or other topological changes. Two problems with topological changes are the geometrical changes and the changes in the soft tissue model caused by the geometrical changes. Most importantly some of the precalculations that have been

used to achieve realtime performance of tissue deformations are not compatible with changes in topology because a recalculation of the precalculated data is too slow for realtime demand.

Basic instrumentation is discussed in this thesis for the interaction with the soft tissue model. Topological changes due to cutting is given special attention. Chapters 7 and8 deal with the topics.

## 3.5 Visualization and Display

Visualization includes standard interfaces to visualization such as OpenGL [55]. Special effects have previously been used to add detail and realism such as blood, smoke and texture to simulations. Also special display devices have previously been investigated, e.g. stereo 3d.

In this thesis the standard graphics platform OpenGL[55] has been used for visualization and the implementation runs on a standard desktop computer with a standard monitor.

### 3.5.1 Usage and Validation

The Usage of the Surgical Simulation is important. As discussed previously I believe it is important to cooperate with surgeons as they are the experts. We need their cooperation to tell us if what we are doing is realistic, and wether we focus on essentials. Many of the recent papers on surgical simulation have been based on a direct cooperation between surgeons and computer scientists. An expert evaluation of my surgical simulator is discussed in chapter 10.

# Part II

# Surgical Simulation

# Chapter 4

# Finite Element Models

The idea of using continuous models of physics for computer animation was introduced by Terzopoulos [75]. Bro Nielsen [58] later used Finite Element Models (FEM) methods for surgical simulation.

Continuous equations that govern the behavior of soft body dynamics can be constructed but are not easily solved. Analytical solutions can be found for simple cases, but for complex cases we must use numerical methods to discretize and solve the problem. Finite Element Models [6] essentially decompose the domain over which the equations of motion are solved.

Finite Element analysis is a general theory of how to solve differential equations over some continuum. In the rest of this thesis we will look only at Finite Element Models as a tool to calculate deformations in soft tissue. We will refer to the Finite Element techniques as FEM.

The main advantage of FEM is that they approximate the actual solution of the equations we set up for the deformation in theory of elasticity. The main problem with Finite Element in the scope of realtime simulation is to make it run fast enough. Several techniques are used to make it run fast. First of all we use some assumptions regarding the tissue under study. Furthermore the solution type considered is static. More complex models exist, but can not be implemented for realtime execution with today's technology.

The idea behind FEM is to divide the continuum of the organ into basic elements over which the differential equations can be solved more easily. The Theory of Finite Element analysis is in itself a large field. Bro Nielsen introduces Finite Element strain analysis in relation to surgical simulation in [58]. Bro Nielsen was one of the first to propose a FEM based deformation of organs for use in surgical simulation.

FEM have been used extensively in simulations demanding correct and very precise solutions. Realtime solutions have been considered unrealistic for quite some time.

FEM has been used in e.g. craniofacial surgery [38] to simulate tissue response due to movement of bone or boneparts in the face. The liver has been effectively simulated in [16]. In [53] the tissue of the arm is simulated as a three

layer model with distinct physical characteristics. Brain surgery has been simulated in [32] using FEM to represent nerves and blod vessel with high precision.

Sections 4.1 to 4.4 are based mostly on [**?**, **?**, 24]. I will not define FEM generally, only present it in sufficient detail to understand how an implementation is built and where issues regarding realism and realtime deformation lies.

Initially I will give a short overview of the sections of this chapter. To find the deformation of an organ under some load we would like to minimize some notion of energy, this energy measure is derived from the Theory of Elasticity, presented in section 4.1. The energy function is defined in section 4.2. To actually solve this energy function we use FEM to discretize the function in section 4.3 and set up a set of linear equations to solve in section 4.4. Numerical techniques can be used to solve the linear equations as presented in section 4.5.

## 4.1 Theory of Elasticity

Continuum mechanics deal with the prediction and calculation of the effect of applying an external load on some body with physical characteristics. The Theory of Elasticity [67] is the part of Continuum mechanics that deals with elastic materials. That is, materials that returns to their original configuration when the external load is released. A body covering a continuous region is discretized to a collection of connected points approximating the shape of the body.

When studying the relationship between forces and deformation, some of the concepts we need to define are stress, strain, equilibrium and displacement [79]. *Stress* is the strength of the force from interactions such as stretching, squeezing or twisting. Often stress is characterized as "force per unit area". *Strain* is the resulting deformation. The stress/strain relationship defines how tissue deforms under a given force. When forces are applied to the tissue it deforms to a configuration of points in which the energy of the tissue is in *equilibrium*. The information about the tissue we would like to know is the *displacement* of the nodes in equilibrium.

The displacement vector logically consists of two different kinds of displacements: The rigid component and the strain. The rigid component is the displacement that is experienced if we assume the distance of all points in the model to be constant. The information we will find is the strain component of the displacement vector.

The simplest model of static reversible elastic deformation is the *linear elastic* model. In this model the Stress/Strain relationship is assumed linear [19], see figure 4.1. The linear relationship between stress and strain is often used in surgery simulation. The behavior of real tissue can be represented by a linear model if the displacement is relatively small (below 10 % of the mesh size)[15]. Linear elasticity has been found experimentally to be valid for small deformations. This is one of the observations of the tissue in children's hearts from section 2.6. As linear model is only valid for small displacements, larger displacements demand more complex non-linear models to be used.

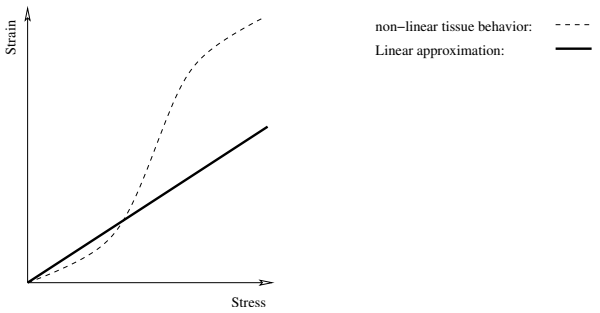



Figure 4.1: The linear relationship between stress and strain approximating a non-linear relationship.

The material properties considered in this thesis are restricted to homogeneous, isotropic, linear elastic materials. Other more complex material behavior exists, such as plasticity (where strain does not return to zero after a cetain stress amount) or viscous material (where the deformation depends on the history of the stress on the material). Also more advanced models including non-linear stress/strain and incompressible volumes can be formulated, but it is not realistically solved in real time with support for topological changes with todays computing power. An overview of some of these material properties is presented in [**?**].

The Finite Element analysis combined with the linear elasticity elegantly lead to systems of linear equations that can be solved relatively fast with a range of standard methods.

In Theory of Elasticity the organ $\Omega$ consists of nodes with an initial position $x_i = [x, y, z]^T$ where $x_i \in \Omega$. Each node also defines a displacement $u_i(t) = [u, v, w]^T$. A node can be either fixed or free. The nodal position of a free node at each timestep is defined as $x_i + u_i(t)$, a fixed node $i$ is always in position $x_i$.

## 4.2 The Energy Function

The potential energy of a system is

$$\Pi \quad = \quad E_{strain} - W$$

Where $E_{strain}$ is the strain energy and $W$ the work done by external forces. The potential energy $\Pi$ reaches a minimum when the derivative $\dot{\Pi}$ is zero, this is the equilibrium that we seek.

The work $W$ is defined as:

$$W \quad = \quad \int_{\Omega} f^T u \, dx$$

The strain energy of the linear elastic body $\Omega$ is defined as:

$$E_{strain} = \frac{1}{2}\int_{\Omega} \varepsilon^T \sigma dx$$

where $\varepsilon$ is the stress vector and $\sigma$ is the strain vector.The stress vector $\varepsilon$, indicating stress displacement relationships [53], is defined as $\varepsilon = Bu$ where

$$B = \begin{bmatrix} \frac{\delta}{\delta x} & 0 & 0 \\ 0 & \frac{\delta}{\delta y} & 0 \\ 0 & 0 & \frac{\delta}{\delta z} \\ \frac{\delta}{\delta y} & \frac{\delta}{\delta x} & 0 \\ \frac{\delta}{\delta z} & 0 & \frac{\delta}{\delta x} \\ 0 & \frac{\delta}{\delta z} & \frac{\delta}{\delta y} \end{bmatrix}$$

The strain vector $\sigma$ is defined in relation to the stress vector $\varepsilon$ through Hooke's law:

$$\sigma = C\varepsilon$$

That is, we have defined a linear stress/strain relationssship. $C$ is the material matrix. Assuming a homogeneous and isotropic material, the matrix is defined by the two Lamé material parameters $\lambda$ and $\mu$:

$$C = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix}$$

Often the material parameters are expressed in terms of Young's modulus $E$ and the Poissons ratio $\sigma$ connected to the Lamé parameters through:

$$\lambda = \frac{\sigma E}{(1+\sigma)(1-2\sigma)}$$
$$\mu = \frac{E}{2(1+\sigma)}$$

Intuitively Young's modulus represent the stiffness of the material and Poissons ratio the compressibility. The closer $\sigma$ is to 0.5 the more incompressible the material is.

The energy function we use is then:

$$E(u) = \frac{1}{2}\int_{\Omega} u^T B^T C B u \, dx - \int_{\Omega} f^T u \, dx$$

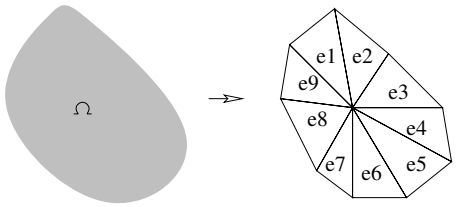



Figure 4.2: Discretization of the shape Ω into triangle elements e1 to e9.

## 4.3 Discretization of the Energy Function using Finite Elements

To find the equilibrium we will discretize the continuum into elements joined at node points, see figure 4.2. We will choose an element type and an interpolation function of the nodes of the elements.

The Finite Elements most often used is the tetrahedral element with linear interpolation of the displacement fields of the four corner nodes. Through meshing, the shape Ω has been discretized into a number interconnected tetrahedrons, see figure 4.2. Inside a tetrahedron we can estimate the displacement by a weighted average of the displacement of the four nodes in the tetrahedron.

$$u(p) \quad = \quad \sum_{i=1}^{4} N_i^e(p) u_i^e \quad \text{with } p = [x, y, z]$$

$u^e = [u_1^{eT}, u_2^{eT}, u_3^{eT}, u_4^{eT}]^T$ being the compound displacement vector, for an element $e$ with nodes numbered $i = 1, 2, 3, 4$. $N_i^e(p) = \frac{1}{6V^e}(a_i^e + b_i^e x + c_i^e y + d_i^e z)$ is the natural coordinate system of the tetrahedron $e$ with $i = 1, 2, 3, 4$ indicating numeration of nodes.

Using this definition of $u(x)$, We would like to discretize $Bu$ to be able to evaluate the expression. Using the definition of of $u(p)$ the components of $B$ (derivatives of $u$ ) become:

$$\begin{aligned}
\frac{\delta u}{\delta x} &= \sum_{i=1}^{4} \frac{\delta N_i^e(p)}{\delta x} u_i^e \\
\frac{\delta u}{\delta y} &= \sum_{i=1}^{4} \frac{\delta N_i^e(p)}{\delta y} u_i^e \\
\frac{\delta u}{\delta z} &= \sum_{i=1}^{4} \frac{\delta N_i^e(p)}{\delta z} u_i^e
\end{aligned}$$

If we look at the definition of $N_i^e(x)$ we see that e.g. in the derivation with respect to $x$ only $b_i^e$ will remain from within the parenthesis, that is:

**Algorithm 1** Creating the global stiffness matrix from element stiffness matrices

```
for (all tetrahedrons e)
    for (the three nodes u of e)
        for (the three nodes n of e)

            K[n.globalNo,u.globalNo]+=Ke[n.localNo,u.localNo]
```

$$\begin{aligned}\frac{\delta N_i^e(p)}{\delta x} &= \frac{1}{6V^e}(b_i^e)\\ \frac{\delta N_i^e(p)}{\delta y} &= \frac{1}{6V^e}(c_i^e)\\ \frac{\delta N_i^e(p)}{\delta z} &= \frac{1}{6V^e}(d_i^e)\end{aligned}$$

We can now replace the derivatives of $u$ in $B$ and rewrite the strain energy, to the discretized strain energy:

$$E(u) = \frac{1}{2}\sum_e \int_{V^e} u^{eT} B^{eT} C B^e u^e \, dx$$

where transformed $B^e$ is a constant matrix that only depends on the shape of the tetrahedron, see appendix A.2. Because everything inside the integral sign is constant, the discretized strain energy reduces to:

$$E(u) = \frac{1}{2}\sum_e u^{eT}(B^{eT} C B^e V^e) u^e$$

where $V^e$ is the volume of the element $e$. $B^{eT} C B^e V^e$ will be defined as $K^e$and called the element stiffness matrix. Some characteristics of $K^e$ is that it is symmetric (see appendix A.1) and positive definit (per definition because $E(u)$ is an energy function).

### 4.3.1 Creating the global stiffness matrix K

We need to assemble the global stiffness matrix $K$ representing the entire mesh from the element stiffness matrices. We transfer the local node numbering to a global node numbering:

We can see that an index in $K$ depends only on the tetrahedrons incident to the nodes that the index represent. This will be used later to enable fast updates of $K$ when the topology of the mesh changes.

## 4.4 Finding the minimum energy configuration

The deformation we seek is the minimal energy configuration of the nodes, that is when $\delta E(u)$ is 0. We will rewrite $E(u)$ in terms of multiplications.

$$\begin{aligned} E(u) &= \frac{1}{2}u^T K u - f \cdot u \\ &= \frac{1}{2}\sum_{j=1}^{3N} u_j \sum_{i=1}^{3N} K_{ji}u_i - \sum_{j=1}^{3N} f_j \cdot u_j \end{aligned}$$

We observe that

$$\frac{\delta u_j}{\delta u_k} = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases}$$

The derivation of $E$ with respect to $u_k$ is:

$$\frac{\delta E}{\delta u_k} = \frac{1}{2}\sum_{i=1}^{3N} K_{ki}u_i + \frac{1}{2}\sum_{j=1}^{3N} u_j K_{jk} - f_k = 0$$

As noted in the previous section $K$ is symmetric, and the equation therefore reduces to:

$$\sum_{i=1}^{3N} K_{ki}u_i - f_k = 0$$

Formulating this for the entire system of equations we get:

$$Ku = f$$

This is a system of 3N unknown displacements, where N is the number of nodes. The matrix $K$ is sparse because (as noted in 4.3.1) the entrances in the matrix $K$ related to a given node are only non-zero where the nodes indicated by the row and column index have a connection to the original node. Standard systems for solving linear systems of equations can be chosen to solve this system.

## 4.5 Solving the linear system of equations

A range of standard methods exist to solve the system of linear equations $Ku = f$; Gaussian elimination, Cholesky Factorization or conjugate gradient to name a few [39, chapter four].

The linear system presented is singular, meaning that no unique solution exists. In the domain of deformation in the three dimensional space this fact has an intuitive explanation. Without any positions prescribed, the body has no unique position in space, and because there is no unique position there is no unique deformation.

To make the system non-singular, we would like to prescribe a number of displacements. $Ku = f$ written out:

$$\begin{bmatrix} k_{1\,1} & k_{1\,2} & \dots & k_{1\,3N} \\ k_{2\,1} & k_{2\,2} & \dots & k_{2\,3N} \\ \vdots & \vdots & \ddots & \vdots \\ k_{3N\,1} & k_{3N\,2} & \dots & k_{3N\,3N} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{3N} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{3N} \end{bmatrix}$$

If we wish to prescribe a displacement of a degree of freedom $u_k = \Delta_k$, we would rewrite the above equation assuming $k = 2$ as an example:

$$\begin{bmatrix} k_{1\,1} & k_{1\,2} & \dots & k_{1\,3N} \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ k_{3N\,1} & k_{3N\,2} & \dots & k_{3N\,3N} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{3N} \end{bmatrix} = \begin{bmatrix} f_1 \\ \Delta_k \\ \vdots \\ f_{3N} \end{bmatrix}$$

To re-create $K$ as a symmetric matrix, we subtract multiples of the second row:

$$\begin{bmatrix} k_{1\,1} & 0 & \dots & k_{1\,3N} \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ k_{3N\,1} & 0 & \dots & k_{3N\,3N} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{3N} \end{bmatrix} = \begin{bmatrix} f_1 - k_{1\,2}\Delta_k \\ \Delta_k \\ \vdots \\ f_{3N} - k_{3N\,2}\Delta_k \end{bmatrix}$$

At least three nodes (or nine degrees of freedom) need to be prescribed to solve the system. Intuitively this amounts the number of nodes that must be held fixed for the body to have a unique position in space.

### 4.5.1 Fast solving

One technique for solving the system is to explicitly invert the matrix $K$ as proposed by Bro Nielsen in [58].

$$Ku = f \Leftrightarrow u = K^{-1}f$$

In Bro Nielsens setup the cost was $O(N^3)$ for inverting of $K$, giving a long pre-computation but interactive update rates [1]. $K^{-1}$ is a dense matrix, but a selective matrix vector multiplication with a sparse force vector can give interactive rates. Unfortunately this solving method is not compatible with changes in topology. $K^{-1}$ cannot easily be updated when $K$ changes.

[1] Bro Nielsen reports interactive rates for up to 250 nodes in 1996, the basic heart model used is 35000 nodes.

Inverting a matrix introduces a significant numerical error, but according to [58] the speed-up factor is around ten times compared to most other methods. Considering our goal of realtime changes in the topology, inverting $K$ is problematic. Re-computing or updating the matrix is not possible in real time [58].

Some methods exist to make $K$ smaller by condensation; substitutting boundary conditions in $K$ thereby solving for forces on the surface of the organ [56]. Again this can give a dramatic speed-up, but is not useful when we want to support topological changes.

### 4.5.2 Supporting realtime cuts

As explained in section 2.6 surgeons often work within a certain area of the organ for some length of time. For a technique known as Region-of-Interest used with FEM the assumption is made that the surgeon only works inside this restricted area for the duration of the simulation. The general idea is to define two FEMs; a slow one supporting cuts and a fast one not supporting cuts. The slow FEM will be used for the region-of-interest and the fast one for the rest of the organ. This approach has been used in e.g. [14] and [32]. The method was used to define a Dynamic FEM supporting cuts within the region-of-interest and a fast static FEM everywhere else. The user will therefore experience that the organ will respond realistically to cuts and interaction in the defined area, and will behave less realistically outside this area.

In our case we would like to support the whole surgical procedure, with shifting regions-of-interest, and we will need a method that supports cuts on the entire surface. When the number of nodes in the organ increases, $K$ increases in size. For large systems direct methods are often not feasible in realtime. A different approach is to use iterative algorithms which exploit the fact that $K$ is sparse. Another argument against direct methods that use pre-calculation of some sort, is that changes in the topology leads to changes in the matrix $K$. This means that the pre-computed data would need to be recalculated, which would not be compatible with the realtime demand.

In [61] the Conjugate Gradient method is proposed as a well suited method for solving the linear equations of the FEM, whilst allowing realtime alterations of the topology. For details of the Conjugate Gradient see [39] and the introduction by Shewchuk [72]. The iterative methods compute a sequence of approximations, $\{x^1, x^2 \ldots\}$ to the solution $x$, converging to the real solution. Many iterative methods are of the general form:

$$x^{(k+1)} = x^{(k)} + t_k v^{(k)}$$

$v^{(k)}$ is the search direction and $t_k$ the distance we move from $x^{(k)}$ to $x^{(k+1)}$. The idea behind Conjugate Gradient is to compute a set of orthogonal[2] search directions $\{v^{(1)}, v^{(2)}, ..., v^{(n)}\}$ and optimal distances $t_k$ for each $v^{(k)}$ such that a step along $v^{(k)}$ of length $t_k$ will line up with $x$. After n iterations we should

[2] actually A-orthogonal.

**Algorithm 2** Conjugate Gradient method

```
r=b-Ax
v=r
c=r·r
for(k=1 to M)
    if (v·v)^½ < δ then exit loop
    z=Av
    t = c / (v·z)
    x=x+tv
    r=r-tz
    d=r·r
    if (d<ε) then exit loop
    v=r+(d/c)v
    c=d
    return x
```

hereby reach $x$. The Conjugate Gradient method was originally introduced as a direct method, but accumulated round-off errors occur to such a degree that the Conjugate Gradient is not used as a direct method. The Conjugate Gradient is very effective as an iterative method though.

A nice property of the Conjugate Gradient method is that it can be initialized with a solution guess. Combined with the fact that the interaction forces often do not change very much from frame to frame, we can effectively seed the Conjugate Gradient at timestep $t$ with the solution found at time step $t-1$. For large models Conjugate Gradient often does not find the solution within the time allowing for realtime simulation. In this case one can choose to show the approximate solution the algorithm has found at the time of abruption, and continue the search seeded with this approximate solution the next frame. The model will then exhibit a behavior that mimics that of a dynamic system, jittering towards the minimum energy configuration. This jittering should not be mistaken for the animation of a dynamic model. It is only due to the visualization of approximate solutions.

As can be seen from the Conjugate Gradient (Algorithm 2), the dominating part of the computation is the sparse matrix vector multiplication. The matrix does not need to be explicitly stored, but can be computed directly from a node and its incident tetrahedrons (see section 4.3.1).

The Conjugate Gradient converges linearly by a constant factor every iteration. This factor depends on the condition number of the matrix $K$, the larger the condition number, the slower the convergence. According to [21] there are no theoretical results connecting characteristics of the mesh to matrix characteristics from which convergence can be deduced. Instead [21] empirically establishes certain relationships between mesh size and quality, and the convergence behavior. A tetrahedron is of a good quality if the tetrahedrons are of approximately the same size and do not have small angles. When the size of the

finite element mesh increases the condition number (and thereby the number of iterations) increase. When the quality of the mesh decreases the condition number increases. Mesh improvement techniques such as mesh swapping and mesh smoothing can be used to improve quality.

Material properties also determine the degree of convergence [61]. As the Poisson ratio approaches to 0.5 the material becomes incompressible - at 0.5 the matrix becomes singular and an extra variable, pressure, must be introduced. the closer the Poisson ratio is to 0.5, the larger the condition number.

# Chapter 5

# Spring Mass Models

We will now look at another elastic model, the Spring Mass model that has often been chosen, when realtime performance was important. The major difference compared to the FEM is that the Spring Mass Model is a discrete model in its basic definition. Both models are of course discrete when we actually compute the resulting deformation, but the FEM is an approximation of a continuum where the Spring Mass model is discrete from the beginning.

The FEM as described in chapter 4 is a static model, while the Spring Mass model is introduced as a dynamic model. That is, a model that exhibits time dependent movement of its nodal points resulting in waves and vibrations.

As with the FEM we represent the organ $\Omega$ with a finite number of nodes. The Spring Mass model is a special particle system. A particle system consists of a number of particles or nodes moving in space under the influence of external forces such as gravity, repelling forces, attractive forces or collision response. Particle systems have often been used to simulate natural phenomena such as smoke or fire. The Spring Mass model is essentially a particle system with a fixed topology connecting neighboring particles with springs that introduce repelling and attractive forces into the system to constrain the shape.

The Spring Mass system is often used in favor of FEM because it can easily run in realtime. The Spring Mass model has mostly been used in realtime applications for training scenarios. The first use of the Spring Mass model for surgical simulation was in Cover et. al. [17] for laparoscopic gall-bladder surgery. The entire abdominal region has been simulated in [57], and specific simulations have been made of the liver [12] and gall-bladder [17]. Hysteroscopy has been evaluated as a case study in [49] with very positive subjective results.

## 5.1 Spring mass formulation

The Spring Mass model introduces two concepts to model the elasticity of an organ: Springs and Particles. An organ $\Omega$ is defined as a number of particles $x_i \in \mathbb{R}^3$ where $x_i \in \Omega$. The particles represent mass and inertia but have no

volume. The Spring's forces are connections between two particles that affect the particles with forces based on their distance.

In this chapter the general notion of node will be interchangeable with the notion of a particle, indicating a physical particle. The notion of an edge will be interchangeable with a spring, indicating transfer of energy.

The position of a particle in space is governed by Newtons second law of motion:

$$f = ma \tag{5.1}$$

where $f$ is force, $m$ is the mass and $a$ is acceleration, that is the second derivative of position $x$.

A spring connects two particles and adds force to the particles based on their distance. Often linear springs following Hook's law are used. A Hookean spring gives a linear relationship between forces exhibited on the particles and the difference between the resting distance and the actual distance of the particles.

To simulate such forces as air resistance and loss of energy in the system, the concept of damping is introduced. Another use for the damping factor is to help ensure convergence of the numerical solutions. We assume that we have $n$ particles that approximate the shape of the organ $\Omega$ and $i, j \in \{1, 2...n\}$. With damping the behavior of the spring mass system is governed by the following equation:

$$m_i \ddot{x}_i = -y_i \dot{x}_i + \sum_j g_{ij} + f_i \tag{5.2}$$

This second order differential equation controls the position $x_i \in \mathbb{R}^3$ of a particle $i$ with mass $m_i$. A velocity dependent damping is introduced to the system via the $y_i$ constant; the faster the particle goes the more energy the system loses due to damping. Often Spring Mass systems are damped beyond a realistic amount to increase stability of the system.

Two different categories of forces act on the particle, external and internal. External forces are forces that are external to the organ, e.g. user interaction and gravity, $f_i$ represent the total external force on the particle $i$. Internal forces originate from within the organ, in the spring mass simulation they are represented by the springs. $g_{ij}$ represents the internal forces as described by the spring between $i$ and $j$. (In an actual implementation the spring is not present when $g_{ij} = 0$ ). For a linear spring $g_{ij}$is defined as:

$$g_{ij} = k_{ij} \left( l_{ij} - ||x_i - x_j|| \right) \frac{x_i - x_j}{||x_i - x_j||}$$

That is, $g_{ij}$ is the vector between the rest and actual configuration of the spring multiplied by the spring-stiffness multiplied by the spring stiffness $k_{ij}$ between nodes $i$ and node $j$, $l_{ij}$ is the original length of the spring between $i$ and $j$. Figure 5.1 shows $g_{ij}$ without the $k_{ij}$ factor in compressed and stretched states. Intuitively, the spring adds attractive forces to the particles if they are further away than nominal distance and repulsive forces if they are closer than nominal distance.

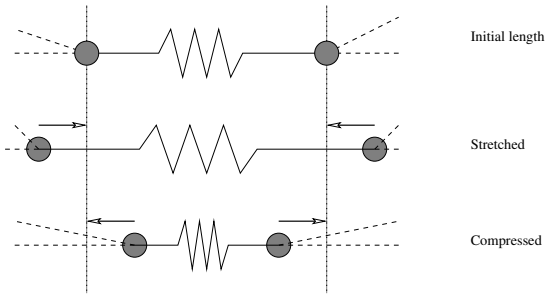



Figure 5.1: Spring response

The Spring Mass model is said to be local because each particle can only react in response to the behavior of the particles that it is connected to through springs. For the dynamic system this means that forces propagate through the organ along the springs, and can only propagate one spring each discrete timestep.

The next two chapters deal with the different kinds of springs and their connectivity. The different spring types and their connectivity are essential for the actual behavior of the spring mass system. Section 5.2 deals with different spring types and section 5.9 deals with spring topology issues.

## 5.2 Internal forces

Several different kinds of special springs have been introduced to fulfill some needs either introduced because of the actual tissue, or because of the level of simplifications made in the approximation of the tissue behavior. In this section we will quickly look at some different force models. For simplicity only the basic linear spring model has been used in the implementation.

### 5.2.1 Home forces

It is a characteristic feature of a Spring Mass model that there is a faster convergence when a few points are grabbed and moved, and a slower convergence back to the initial configuration of points when the points are released. To help the Spring Mass system converge to the minimum, one can introduce springs connecting the initial position to the particles. We know that there is equilibrium in the initial configuration of nodes. In [17] this approach is called home forces. The home forces also simulate volumetric forces simply because the Spring Mass system will return to an initial configuration in which volume is preserved.

Home forces could also be connected to rigid bodies giving both a global (rigid) and local behavior (soft).

Home forces are not usefull when topological changes can be made, because the minimal energy configuration is changed.

### 5.2.2 Nonlinear force models

If $k_{ij}$ is a constant the Spring model is said to be linear, but $k_{ij}$can also be a function of the distance between node $i$ and node $j$ to exhibit nonlinear behavior. In [73] the non-linear stress strain curve of facial tissue is approximated by a by-phasic function as follow:

$$k_{ij} = \begin{cases} k_{ij}^1 & \text{if } l_{ij} - ||x_i - x_j|| \leq e_{ij} \\ k_{ij}^2 & \text{if } l_{ij} - ||x_i - x_j|| \leq e_{ij} \end{cases}$$

In [40] a non-linear behavior is approximated by a third-degree polynomial.

Only linear forces have been used in the implementation presented to the surgeons, because we get a greater stability and linear forces should be a valid approximation for small forces.

### 5.2.3 Volume preservation

In [73] a force that constrains the volume is used to model incompressible materials.

## 5.3 Solving the second order differential equation

From equation 5.2 we would like to determine the position of the particles in the spring mass system to animate the behavior of the system. If we have a particle $x_i$ at time $t$ we would like to know the position of $x_i$ at time $t + \Delta$ assuming we know what forces act on the particle in that period of time.

To solve the second order differential equation governing the position of the nodes in time, one often expresses the equation as two first order differential equations and solves with standard methods such as Euler integration or Runga Kutta. The Verlet method on the other hand is based directly on the second order differential equation. The initial values of the position $x$ are assumed to be given in the rest of this section.

With the introduction of a velocity variable $v = \dot{x}$ the equation 5.1

$$\ddot{x} = f/m$$

is rewritten to:

$$\begin{aligned} \dot{v} &= f/m \\ \dot{x} &= v \end{aligned}$$

The choice of integration method is a trade-off between computation time and precision of the integration.

### 5.3.1 Explicit Euler Integration

The most basic integration formula is the explicit Euler [39, chapter 8]. Euler integration of the ordinary differential equation $\dot{x} = h(x)$ is simply a Taylor-series of order 1:

$$x(t + \Delta) = x(t) + \Delta \cdot h(x)$$

With respect to the second order differential equation 5.1 the solution is:

$$\begin{aligned} x(t + \Delta) &= x(t) + \Delta \cdot v(t) \\ v(t + \Delta) &= v(t) + \Delta \cdot (f/m) \end{aligned}$$

Explicit Euler integration is very simple to compute, but is inherently unstable.

### 5.3.2 Runga Kutta

Runga Kutta is often presented as a more precise and stable alternative to Euler integration, but is also slower to compute, because $h(x)$ must be evaluated several times. Often Runga Kutta 4 is used in e.g. [4]. Runga Kutta 4 reproduces the terms of the Taylor series up to one involving $h^4$. The error is of size $O(h^5)$. Runga Kutta 2 has been used in e.g. [73].

One advantage of the Runga Kutta family of integration is that they support a change of stepsize to increase accuracy of the integration on parts of the function. This feature is not immediately useful in a realtime surgical simulator because we will need a stable flow of frames at all times.

### 5.3.3 Verlet

The Verlet integration is based on two third-order Taylor expansions of the positions x(t) , one backward and one forward:

$$\begin{aligned} x(t + h) &= x(t) + \dot{x}(t)h + \frac{1}{2}x^{(2)}(t)h^2 + \frac{1}{6}x^{(3)}(t)h^3 + O(h^4) \\ x(t - h) &= x(t) - \dot{x}(t)h + \frac{1}{2}x^{(2)}(t)h^2 - \frac{1}{6}x^{(3)}(t)h^3 + O(h^4) \end{aligned}$$

Adding the equations and isolating for $x(t + h)$ gives us:

$$x(t + h) = 2x(t) - x(t - h) + \ddot{x}(t)h^2 + O(h^4)$$

Because we integrate Newton's s equations, $\ddot{x}$ is know directly as $\frac{f}{m}$ 5.1. The Verlet method is reasonably fast to evaluate and is very stable.

The damping of the Spring Mass system was introduced as a linear function of the velocity. In the standard Verlet method the velocity is not expressed

directly, and we will therefore use an ad-hoc method of weighting the old and new positions:

$$x(t+h) \quad = \quad x(t-h)(1-\lambda) + x(t)\lambda$$

Verlet integration was originally introduced in the field of molecular dynamics.Verlet integration is more rarely used in surgical simulation than the standard methods of Runga Kutta and Euler integration. Verlet integration was used in [36] for advanced charachter physics and [51] for surgical simulation.

### 5.3.4 Stability

The integration methods used to solve the differential equations of the spring mass system have a trade-off between numerical accuracy and speed of calculation. We need a realtime solution, and the numerical accuracy is therefore of lesser importance. More important is that the solution is stable.

In [51] Euler, Runga Kutta 4 and different Verlet[1] methods were compared with respect to their ability to deliver stable realtime results. It was tested how large the time step of the individual methods could be set while remaining stable. Taking into account the calculation time of the integration, the Verlet method is superior to Euler and Runga Kutta 4. The Verlet method is therefore chosen as a standard in my Surgical Simulator, although other methods could easily be supported.

Because stability is most important, some measures can be taken to increase it. One method is to dampen the system beyond a realistic level. Relaxation, which will be introduced later, is another very effective way of increasing stability, but the behavior of the system changes dramatically.

I have experienced experimentally that increasing the model size increases the in-stability of the models. This is probably due to the energies traveling in the system being larger, because adding more springs introduces energy into the system. If this energy gets concentrated in parts of the model, e.g. at the interaction points, it might lead to instability. Therefore Stability is especially important for large and complex geometries such as the heart.

## 5.4 Static equilibrium

The standard Spring Mass system introduced in the previous sections is a dynamic system. Each particle has some inertia and mass. When we simulate such a system, we get an animation of the system finding the minimal energy configuration. Energy introduced into the system will result in vibrations and waves. It has been noted in section 2.6 that in the case of surgery, interaction with the tissue is done in such a way that vibrations do not seem visible. Because of

[1] Basic Verlet is not checked, but is equivalent to velocity Verlet when the velocity is not needed

the relatively small movements and the characteristica of the tissue, the state of equilibrium is reached fast. The idea for a static solution for the Spring Mass system is to approximate the dynamic solution of a full spring mass system by a static solution without inertia, mass or damping.

In each timestep we need to find the static equilibrium of forces. The static equilibrium is expressed as:

$$\sum_j g_{ij} - f_i = 0$$

That is, when internal forces and external forces are in "balance" for all particles $j$.

To guarantee realtime performance an iterative algorithm is used although an accurate solution still cannot be guaranteed at each time step. The iterative algorithm is seeded with the previous solution because this might be close to the configuration in the next timestep.

In [7] such a system of equations is solved with an iterative method as the following:

**Algorithm 3** Quasi Static Algorithm

`Repeat until time` $\delta$ `has elapsed`

`for every` $i \in \{1, 2..., n\}$

(a) $F_i = \sum_j g_{ij} + f_i$

(b) $x_i = x_i + \alpha F_i$

For every particle we find the vector representing forces on the particle. The particle is then displaced along this vector. The algorithm looks very much like an Euler integration without the velocity variable. The $\alpha$ constant is in many ways like the step size of a numerical solution of a differential equation. It must be low enough for the algorithm to converge, but as big as possible for realtime performance. [7] defines this constant experimentally. In chapter 9 we find an optimal $\alpha$ and spring stiffness by optimization against a FEM model.

The algorithm guarantees a given framerate through the $\delta$ constant, but is not guaranteed to reach the actual equilibrium state. Implicitly the algorithm is seeded with the positions of the nodes at the previous timestep, and we can take advantage of the fact that our interaction only causes small changes each time step.

## 5.5 Point interaction

We can make the assumption that the only forces influencing the tissue are concentrated in few areas of small size - we will call this *point interaction*. Together with the assumption that the deformations are of local nature, we can restrict the degree of calculation necessary to calculate the deformations.

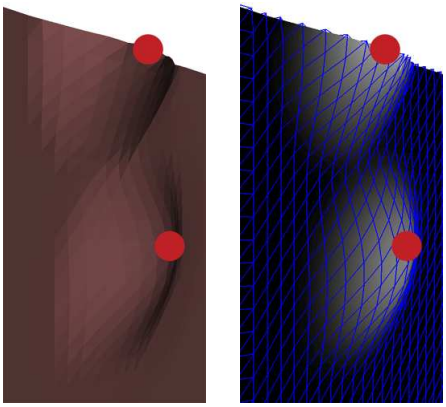


Figure 5.2: Local interaction and deformation at the two red points.

Due to the assumption about point interaction we will not represent gravity, because gravity will by its nature affect all points in the model. Leaving out gravity seemed realistic to the surgeons, because gravity does not have a great effect on surgical procedures.

Because we know that interactions happen in points (or a collection of points close together) and the deformation is local, we can define a range in which we are interested in the deformation, see the illustration in figure 5.2.

The Quasi Static algorithm is especially suited for this kind of interaction. Because step b of the Quasi-Static algorithm 3 uses the most recently computed position of the adjacent nodes, a breadth-first run through the particles from the interaction point has the potential to converge faster.

We make a breadth-first traversal of the nodes in the organ from the interaction point. Nodes are attributed a level that tells us which level of the breadth-first traversal the node is in. The (re-)ordering of the nodes is done each time an interaction is issued on the organ, e.g. when we grab the organ somewhere.

In [7] it is proposed to stop the breadth-first traversal (cut-off) at a given level and effectively define a range in which deformation is calculated. The level at which to stop is found when the difference between old positions and new positions on a given level is lower than some threshold. For large forces this unfortunately means that we might have to run through the entire model, and this cannot be done in time for a realtime response. To accommodate the realtime demand in my implementation, the level of breadth traversal is constant through one run of the simulation.

A problem with the cut-off and re-ordering is that some particles might get stuck if they are still converging from an old interaction when a new ordering of particles is initialized. When a re-ordering of nodes is issued some particle that where previously inside the active area might now be outside. If these particles

**Algorithm 4** Relaxation

```
Repeat until time δ has elapsed
    for all(edges e)
        if (actual length of e is too short)

            delta = abs(actualLength,relaxedLength)
            displace node 1 of e delta towards node 2
            displace node 2 of e delta towards node 1
        else if(actual length of e is too long)


            delta = abs(actualLength,relaxedLength)
            displace node 1 of e delta away from node 2
            displace node 2 of e delta away from node 1
```

had not reached equilibrium, they will stay in their position out of equilibrium until they are in an active area again. One solution might be to combine the previous ordering with the newly issued one and discard the old ordering when all particles in the old ordering are at equilibrium.

When dealing with very big (and complex) models like the cardiac model of this thesis, the cutoff method is very successful because the size of the model has less influence on the performance of the elastic simulation.

In [7] an analysis of the error introduced by larger models on the convergence to equilibrium was made. It was found that for larger objects the error did not continue to grow. The interpretation was that the assumption of local deformation (in a spring mass model) is true.

## 5.6 Relaxation

In [66] the phenomenon of elongated springs was identified in simulation of cloth. Elongated springs are identified as a system in which some springs are stretched unrealistically in relation to others. We would like the springs to be of approximately equal length. One solution is to use a technique known as relaxation [66, 36], in which the springs are iteratively transformed to their relaxed configuration, see algorithm 4.

The nodes are not displaced until the difference between the initial length and the current length is greater than the *linear factor*. That is, we defined the relaxed length as:

$$l_{relax} = \begin{cases} l + linearFactor \cdot l & \text{if } (l_a > l + linearFactor \cdot l) \\ l - linearFactor \cdot l & \text{if } (l_a < l - linearFactor \cdot l) \\ l & \text{else} \end{cases}$$

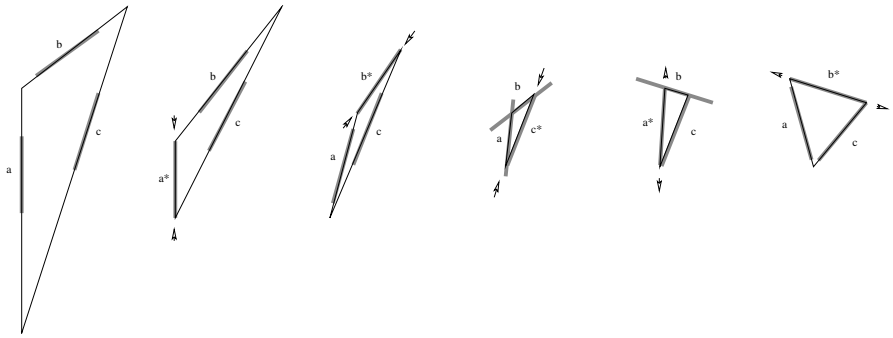



Figure 5.3: Relaxation on a deformed triangle

where $l_a$ is the actual length of the springs, and $l$ is the natural or initial length of the springs.

An example of a deformed triangle being realaxed can be see in figure 5.3. The relaxation helps deformation propagate quickly through the material.

Other kinds of relaxation have been used in connection with the 3d chain mail algorithm [28] where the relaxation is used as an iterative method of finding the minimal energy configuration after a heuristic deformation.

Relaxation has no direct physical interpretation, it is a heuristic model introduced to remove a behavior that did not seem realistic. Relaxation mimics non-linear materials in some ways because it can resist larger forces than the basic linear spring mass model.

One could argue that relaxation only replaces non-linear stress/strain relationships. Although no formal investigation has been made, it is evident that relaxation increases stability of Spring Mass tissue, where we would expect the stability to decrease with the introduction of a non-linear stress/strain relationship.

In an extreme case we might imagine a single tetrahedron with relaxation versus a single tetrahedron with a non-linear stress/strain relationship. It is clear that if we would seek to simulate a near-rigid material a few iterations of relaxation would make the tetrahedron seem rigid. If the non-linear stress/strain relationship should do the same, the step size would have to be very small - increasing the computation time beyond the computation time needed for the relaxation.

For large models we have experienced that the relaxation can increase stability dramatically but at a relatively big computational cost.

## 5.7 Local interaction in large models

As explained in section 2.7, heart geometry is complex and demands a large geometry to represent the complete shape. Spring Mass simulation of the entire

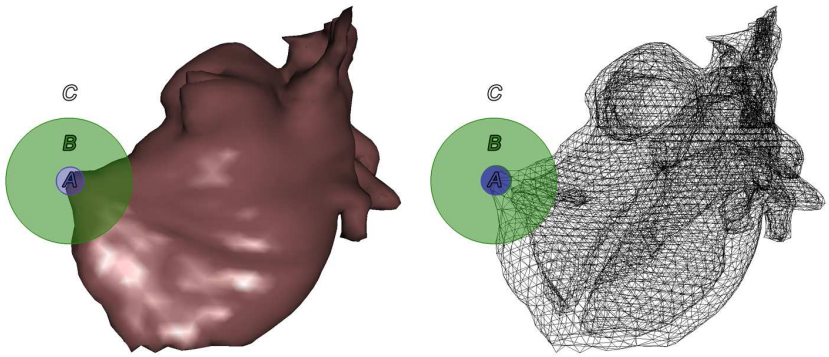



Figure 5.4: Dividing the deformation into an A, B and C area. A area, where the position is set absolute. B area where the position is calculated with Spring Mass and relaxation. C area where position is calculated with relaxation only

model will give us very slow convergence. We use both the fact that surgeons work in regions-of-interest for a length of time, and the fact that the tissue behavior is often local. The same assumption was made in section 4.5.2 with the FEM.

We will make the assumption that the region-of-interest is centered around the current interaction point. The region-of-interest behavior will be governed by the quasi static method with cutoff. To get *some* response from the rest of the organ, we will do a relaxation of the entire organ. The relaxation gives us a faster response from the surrounding tissue than with a pure spring mass simulation of the entire organ. The technique is presented in figure 5.4.

The response from this combined model might not be as realistic as a pure spring mass simulation, but because of the assumption of a region-of-interest we are primarily interested in the local behavior. We will get a faster response, something that can otherwise be problematic for big models.

The model presented in this section will be called a the Local Relaxation Spring Mass ( LR Spring Mass model ). The LR Spring Mass model is an original contribution of this thesis as a combination of previous techniques. This specific model is shown to be better suited than previous techniques for simulation of cardiac surgery, see chapter10.

## 5.8 Hardware speed-up

The Spring Mass method is one of the simplest physically based elastic models that exist. If a physically based elastic model is to run faster one possibility is to use dedicated hardware such as in [5]. Dedicated hardware for physics simulation is not avalible yet, but modern graphics hardware is. Vertex shaders and pixels shaders have become part of a modern 3d graphics hardware, and

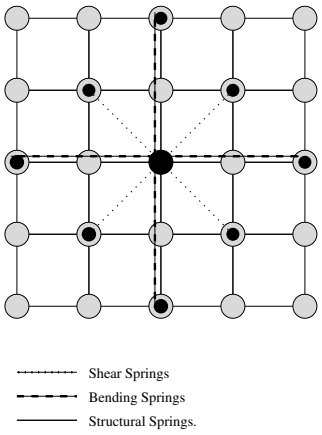



Figure 5.5: 2d regular grid with additional springs

can be used to do physically-based simulation [34]. It was out of scope for this thesis to implement techniques in vertex or pixels shaders, but in future research e.g. the LR Spring Mass model could probably be implemented on a vertex and pixels shader, thereby speeding up the algorithm.

## 5.9 Spring topology issues

The springs represent constraints and flow of energy in the spring mass system, and the topology of these connections as well as the relative position of particles determine the global behavior of the simulated organ. The logic behind the connection of particles also determines the possible visualizations and interaction.

An under-constrained system might have several resting positions and the system might easily end up in configurations of nodal positions which is allowed by the system, but is not realistic. Parts of an under-constrained system can collapse because part of the volume might flip into itself. When a grid structure is used as basis for spring connections the system is under-constrained because the boxes or cubes are in themselves under-constrained. In a 2d grid the missing forces have been identified as missing resistance to shear. The solution is to connect springs across the diagonal. These are called shear springs [66], see figure 5.5 . In a 3d grid composed of boxes additional springs connects corners of the box to resist a collapse [73].

If the system is over-constrained it will exhibit less elasticity and more rigid behavior than we indicated through the spring stiffness. Another problem with an over-constrained system is numerical stability (these kind of stability problems are closely related to the problem of stability and spring-stiffness, see section 5.3.4). Because of these difficulties, systems have often been arranged

in regular structures guaranteeing a homogeneous behavior all over the organ. Often regular lattices or prisms with triangular base have been used.

The strategy behind the connection of nodes through springs is important for the visualization and interaction with the organ. The strategies for connecting the springs are often divided into two categories; surface and volume representations. The surface representation simply defines only a surface with no explicit volumetric elements, the surface model may nonetheless be used to approximate volumetric behavior. The volume representation has elements to ensure a behavior that takes into account the volume of the model. Often the particles of a volumetric representation are arranged in a set of connected tetrahedrons, hexahedrons or other volumetric geometries.

The choice of surface, volumetric or other hybrids, can be regarded as a hierarchy of approximations in which true volumetric models are more precise than surface models. A surface model representing the surface of some volume, will be faster to compute than a full simulation of volume. The trade-off again is efficiency of computation versus physical accuracy [19].

Because the topology of springs is so important for the behavior of the system and the speed of computation, the characteristics of the simulated organ can be taken into consideration when planning the strategy for connectivity. In [12] the behavior of a liver was simulated by two different geometrical components; A 2d elastic surface to simulate the membrane of the liver (with torsion springs to simulate curvature of the surface) and a 3d mesh to simulate the interior of the liver (as a quasi-viscous material)

### 5.9.1 No best way

If the computation power was not a problem, we might ask ourselves what the best topology scheme would be. There is no simple answer to this question. Surely, volumetric behavior is more realistic than surface behavior because a simple surface model cannot preserve volume to the same degree as a volume model. But the spring mass system might not exhibit a more realistic behavior even when the resolution and connectivity of springs resemble some model of reality more closely. If the resolution or detail of the model is increased the behavior cannot be guaranteed to be the same as in the low resolution model. This basically has to do with the fact that a spring mass system is defined locally; there is no global differential equation that we solve for an energy minimal configuration, this is implicit in the spring mass system.

When the resolution is increased the propagation of forces is slower because forces only propagate along one spring each time step. In a higher resolution model the mass also has to be divided in some way, but because of differences in the connectivity this might not be simple.

The answer is that we must validate the models we build, either by comparing them to previous models, formally validate them against real data or get expert opinions [27]. As it was out of scope for this thesis to validate against real data, I have chosen to validate against more precise models and get expert evaluation in chapters 9 and 10.

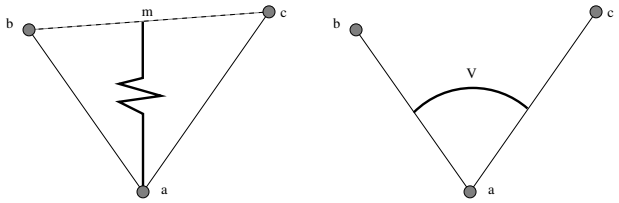



Figure 5.6: Spring to maintain curvature

### 5.9.2 Surface models

When springs are connected in a way that only represents a surface the elastic model is called a surface model. A classic surface representation is the 2d grid with springs to resist shear and bending [66] see figure 5.5. When a basic surface model is used to approximate the behavior of a volumetric organ, there are of course no explicit forces that react to changes in volume. Approximating volumetric behavior of a thin tissue with a surface model covering the surface of the tissue can lead to bad results because the model can easily self intersect. Surface models might in that case be used to model the thin tissue as an infinitely thin 2d grid. Vessels have been simulated in this way, approximating the thin wall of the vessel with a surface model [19]. In [17] a basic surface model has been used to represent a gall-bladder.

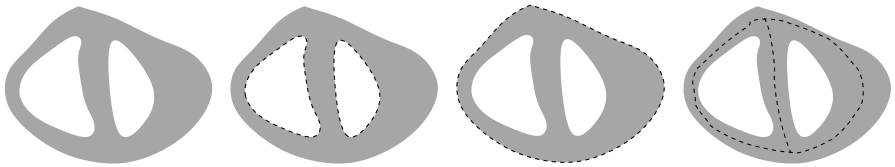
To get some volumetric behavior additional springs are sometimes introduced into the surface based models. The basic approach is to introduce springs to guarantee some curvature of the organ. If the surface consists of a simple 2d grid, bending springs can be introduced to resist bending of the surface, see figure 5.5. In [54] the animation of a muscle along an action line is simulated. The muscle geometry is organized in a grid along the surface of the muscle. [54] therefore introduces angular springs (see figure 5.6 (a) ) to preserve the volume and the overall shape of the muscle. In [12] torsion springs (see figure 5.6 (b) ) are added to maintain the curvature of the surface. All these different springs reduce to a computation over two of three springs arranged to maintain the curvature of the surface.

The surface model is easy to use for the heart because the data we get from the segmentation is the inner surface of the heart. With a little manual extra work, the outer surface can also be generated. In figure 14 the surfaces of the heart can be seen.

With the need to make cuts into the model, a simple surface model representing the exterior or interior of the heart is not sufficient. As soon as the surface is cut into, the surgeon would see that the heart is empty.

Another approach would be to use both the inner and outer surface to represent the tissue surface of the heart. This surface would quickly collapse after all, because the walls are very thin some places.

The third possibility of approximating the heart with a surface model would



(a) Volumen indicated by the two surfaces from segmentation.

(b) Inner surface

(c) Outer surface

(d) Single surface to represent heart wall.

Figure 5.7: Surface models of heart

be to approximate the heart walls with one infinitely thin surfaces. The surface could be calculated from the outer and inner surfaces as a surface in between these two surfaces. The heart wall does not have the same thickness everywhere, and a representation of the wall as one infinitely thin surface would pose a problem both towards the visualization as well as the behavior. The heart would simply not look the way it does in reality. The deformation would also not take into account the varying thickness of the heart wall.

### 5.9.3 True volumetric model

We will define a true volumetric model to be one in which the volume of the entire organ is made up of smaller atomic parts which represent volume. Often such atomic parts are tetrahedra or hexahedral elements.

The tetrahedral mesh gives a great deal of flexibility for an efficient representation of anatomical structures [4], but can be difficult to construct. Tetrahedral elements are structurally stable, but hexahedral elements are under-constrained. When hexahedral elements are used to represent volume extra springs are often inserted across the element [73].

The simple volumetric model can be extended to encompass more elaborate biological features. In [73, 38] a three layer model of the skin tissue is used because the three layers have different physical characteristics.

A mesh of tetrahedra would represent our heart in satisfactory manner. The heart would support realistic cuts, and the thickness of the heart wall would be part of the simulation. The only drawback is that it demands a meshing of the surfaces to create a mesh of tetrahedrons instead of an outer and inner surface. This would require a meshing program, which was not built as part of this thesis. Instead another approach was selected.

### 5.9.4 Connected Surfaces

As part of this thesis a specific volumetric model has been developed. The model is developed with the heart model in mind and requires no special meshing (see section ) - it behaves as though it was a true volume representation. This model

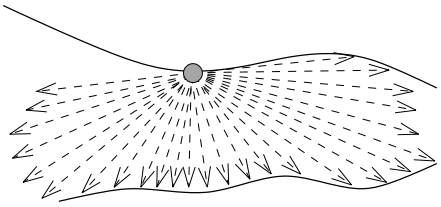


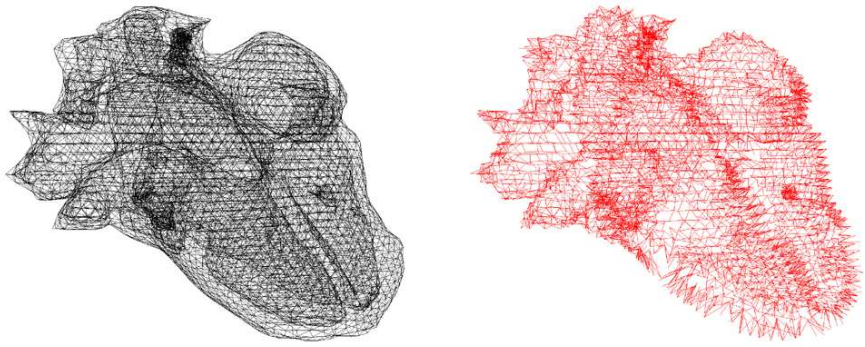
Figure 5.8: Forces between surfaces

is developed as an alternative to the true volumetric tetrahedron or hexahedral mesh of the heart. From the segmentation we get surfaces representing the outer and inner surfaces of the heart muscle. Data from the segmentation is given as a surface consisting of triangles.

The idea of Connected Surfaces is to use both the inner and outer surfaces and set up a relationship between them so that a volumetric behavior is exhibited - the relationship is defined as additional forces. These additional forces, called connecting forces, would propagate force between the two layers to resist bending and constrain the distance between the inner and outer surface. The forces are represented as additional springs connected from each particle to particles in the surface on the opposite side as depicted in figure 5.8. Note that the particles we wish to connect may belong to the same surface, we must just make sure that connections are not made that already exists in the surface (no double forces).

Because we only connect a rather small finite number of springs from each particle we must carefully select the most important springs to connect. When deciding on the scheme we want to use for connection of forces we must also decide on the actual number of connections per node. If we had a rather large number of connections per node we could spread them randomly in a hemisphere above the node in the opposite direction of normal. The spring stiffness of a single connection could be adjusted according to the volume that it represents in the hemisphere (the idea being that a greater volume of tissue gives greater resistance to forces in that direction). When we have a rather small number of connections this scheme might result in under constrained surfaces, resulting in dangling nodes. The solution is to use a scheme that connects the nodes in a regular manner.

We seek a volumetric model that is comparable to the tetrahedron mesh and the constant number of three connections per node is selected. The observation that leads to the actual connection scheme is that the connecting forces along the opposite of the surface normal of a particle have the greatest importance for the behavior of the organ. These forces directly determine how well the organ preserves its volume with respect to the distance between the surfaces, which is often the most difficult part. Connecting forces that have a greater angle off the opposite of the surface normal determine how well the organ resists bending, but often bending happens over a greater area, and several connecting forces with a smaller angle off the opposite of the norm can resist bending just as well.



(a) Surface springs (b) Springs connecting surfaces

Figure 5.9: ConnectedSurface for the Heart

The scheme used is to find the closest triangle in the opposite direction of the norm of a given particle and connect all the three particles in the triangle with the given particle. To speed up the search an acceleration structure for the ray-triangle intersection has been used.

The Connected Surfaces scheme used for connection of the inner and outer surface of the heart can be seen in figure 5.9. Connected Surfaces is an original contribution of this thesis to easily create a volumetric model from surfaces.

## 5.10 Evaluation of Spring Mass in comparison with FEM

We will now look at the behavior of the Spring Mass model in comparison with the Finite Element model.

Often Spring Mass models have been selected in favor of FEM because they can immediately be animated at interactive rates. In recent articles it has been shown that certain FEMs deliver as fast a framerate as Spring Mass models. Traditionally though, FEM have not been used for realtime applications.

When it comes to realism of behavior, the Spring Mass model is a significant approximation of real physical behavior. The primary reason behind this is that the Spring Mass system introduces a discretization into particles and springs, and secondly their behavior. In FEM the behavior of the system is defined first, and the discretization is a method to actually calculate the behavior. The Spring Mass system per definition has only local information for decision and calculations. The Spring Mass system consequently reacts quickly to local changes, but more slowly to global changes. The FEM does not in itself favor global or local changes although methods for solving the system might.

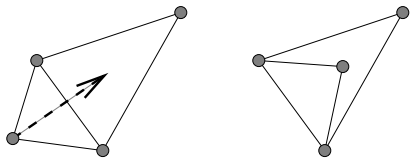


Figure 5.10: If enough force is introduced on one of the two nodes that are only a member of one tetrahedron, it might travel to the other side of the opposite triangle and effectively collapse volume with no forces to counter the change of shape for the entire body of tetrahedrons

The effect of local information on the behavior of the tissue can be severe. Because the system has no notion of the shape of the entire organ irregularities in the topology can occur if large forces are introduced into the system, see figure 5.10. The problem can occur because of the discretization made in the integration. This problem is quite severe because not only is the volume and the geometry invalid. A flip can in many cases lead to numerical instability because the flip can lead to added force or sharpened constraints on other nodes of the model. A flip can also lead to other degeneracies of the mesh. More element flips might appear because of a change in the force distribution. One can introduce checks for such situations, but at a significant computational cost. Furthermore the system would have to be repaired, which would be difficult. Alternatively the simulation could be run at a smaller timestep, lowering the framerate and speed of convergence. Furthermore, the integration method would have to support a change of stepsize.

The parameters of the Spring Mass models in comparison to the parameters of the FEM cannot simply be transfered from one model to the other, there is not clear relationship because the very foundations of the models are different.

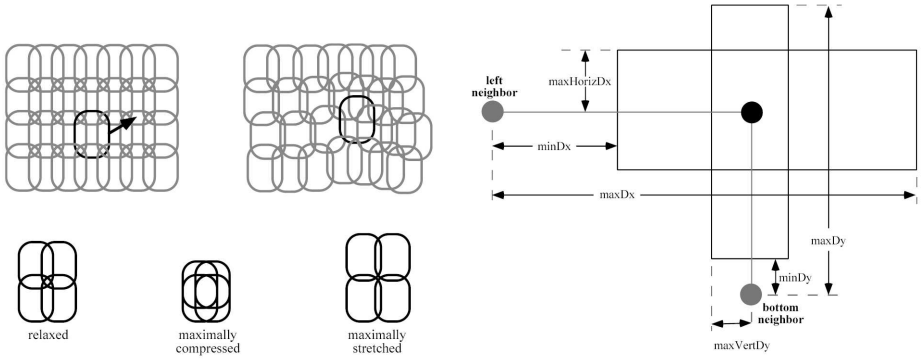
As already discussed, the local information of spring mass models means that the behavior of the model depends on the topology of springs. If one is aware of this fact, it is also one of the advantages of the model, because we can easily determine or define behavior by introducing special springs.

Experiments have shown that Spring Mass and FEM behave almost the same way for small deformations, but for larger deformations they behave very differently. In [38] craniofacial surgery was simulated with Spring Mass and FEM and there was some indication that they behave the same way.

In chapter 9 a comparison between FEM and Spring Mass methods based on optimal parameters is presented.

## 5.11 3D Chainmail

To set the two physical based models in perspective, we will do a quick comparison to an alternative model of deformation. The 3D chainmail [**?**, 23] was introduced by Gibson as a model to favor size of geometry and speed of compu-




(a) Deformation of a 2D chainmail

(b) Node constraints

Figure 5.11: Deformation of chainmail (from [23])

tation. Consequently the deformation is less realistic than Spring Mass model and FEM.

The 3D chainmail algorithm is build as a fast algorithm that can work on large datasets. Using voxel datasets directly allows us to use the information present in the voxels, e.g. density. Furthermore, translation of the datasets to surfaces or other representations, inevitable introduces errors and loss of information [25].

The algorithm assumes a regular structured grid of voxels, which in turn are connected to their six closest neighbors. Each voxel is allowed to move freely within some minimum and maximum distance in each axis according to the positions of the connected neighbors. If the distance constraint is violated, the displacement is transfered to the neighbor voxels depending on the axis on which the constraint is violated, see figure 5.11. As such the algorithm exhibits only plastic behavior in which there are many allowed configurations of voxels.

The basic idea of constraint handling between adjacent nodes is used in both 3D chainmail and iterative relaxation. The area in which the nodes can move freely is often larger in 3d chainmail than in relaxation. Furthermore 3D chainmail is build directly on the grid structure and can resolve the constraints through one loop through the set of nodes. It should be noted here that we are not using relaxation in the surgical simulator for pure resolving of geometrical constraints. We exploit the iterative relaxation combined with Spring Mass based movement of nodes to achieve faster convergence and more realistic deformation.

The 3D chainmail algorithm uses point interaction (as in section 5.2) to guarantee the fast deformation. The deformation is resolved from the interaction point in a breath first manner, comparing each node to its neighbors. As deformation can be absorbed by the area of free movement, there is a good chance that the deformation is retained within a small region.

The basic 3D chainmail does not find a minimum energy configuration of nodes because nodes are free to move within a certain area. The deformation is plastic to some degree, and several configurations of nodes are allowed. In [28] Gibson present different iterative relaxation algorithms to resolve this issue. [65] defines a shape retaining re-formulation of the 3D chainmail algorithm because the basic 3D chainmail has difficulties retaining its shape after an interaction.

In general we can see that the 3D chainmail algorithm is not as physically plausible as the Spring Mass model or FEM; no forces or energy are described in the system. The 3D chainmail algorithm is basically a constraint handling technique for distances between nodes. The resulting deformation cannot be realistic, but the deformation (apart from the iterative relaxation) is resolved very quickly compared to the iterative algorithms of Spring Mass models and FEM.

The chainmail method has been used in e.g. knee Surgery[26]

# Chapter 6

# Design Issues of a Unified Soft Tissue Simulation

As part of this thesis I have implemented a range of soft tissue simulation techniques, interaction techniques and topology altering techniques. These different aspects of a surgical simulation have been implemented in a unified framework to enable sharing of common features and facilitate comparison in the cases of use. The design is constructed in an Object Oriented perspective [44], and presented as UML diagrams [41]. In comparison with the actual implementation the design is naturally not a complete description of the implementation. Several elements have been left out of the UML diagrams for simplicity, and a few elements differ slightly compared to the actual implementation. The general design of the entire simulator is presented in figure 6.1.

A soft tissue simulation can primarily be divided into two phenomena; the geometry/topology of the organ simulated and the computation of deformation/elasticity. The advantage of this assignment of responsibility is that several different ElasticObjects can use several different Geometry objects. In my soft tissue simulator the different Spring Mass models can use Tetrahedron Geometry, Surface Geometry or ConnectedSurface Geometry. The FEM can use the Tetrahedron Geometry.

## 6.1 ElasticObject

The general phenomena regarding an elastic model have been recognized and modeled. The UML diagram is presented i figure 6.2. The behavior of an elastic model can be divided into the behavior of the components that the geometry is constructed of. That is, the elastic behavior is captured in special Nodes, Edges, Triangles and Tetrahedrons. To define a complete soft tissue simulation, specializations of ElasticModel, Node, Edge, Triangle and Tetrahedron classes must be constructed.

To allow the Geometry class to instantiate the specialized Nodes, Edges,

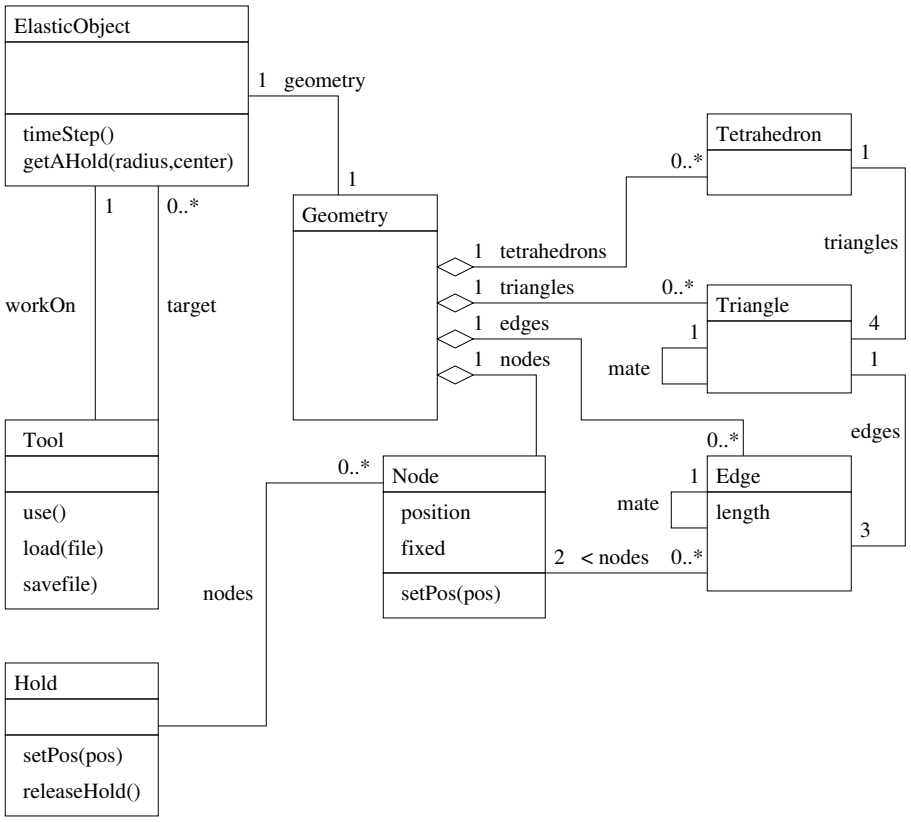



Figure 6.1: UML diagram of general design. The phenomenon of a soft tissue simulator is divided into a Geometry and a ElasticObject. The Geometry class represents the geometry and topology of the soft tissue. It has the responsibility to load file-formats and construct the hierarchi of geometric components. The geometric components used to build the actual geometry are Tetrahedron, Triangle, Edge and Node. The ElasticObject class represents the calculation of deformation and dynamics. Geometry and ElasticObject communicate through a well-defined interface, enabling independent specializations of both classes

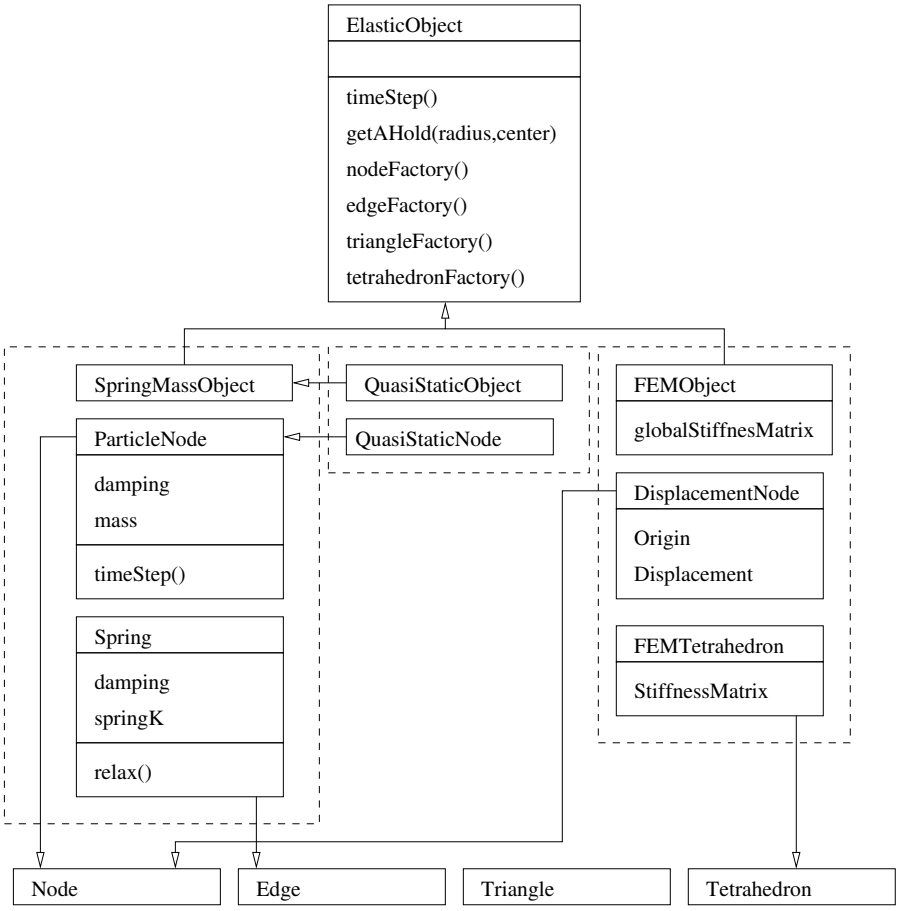



Figure 6.2: UML diagram of a the general elastic model (ElasticObject) and the specialized elastic models: Spring Mass model (SpringMassObject), Quasi Static model (QuasiStaticObject) and Finite Element model (FiniteElementObject). Specializations of Nodes, Edges, Triangles and Tetrahedrons define the geometrical components for use in a specific elastic model

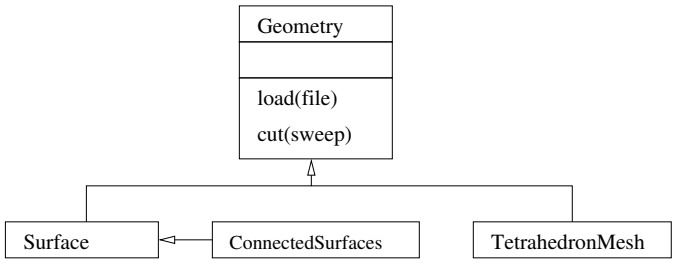



Figure 6.3: UML diagram of Geometries. The three geometrical types are pure surface, connected surface and tetrahedral mesh. All geometrical types can load specific files and make cuts with specialized algorithms

Triangles and Tetrahedrons in the construction of a geometry, virtual factory methods [41, p. 473] are overwritten in the ElasticObject. The Geometry object simply calls the factory methods and receives a specialized Node, Edge, Triangle or Tetrahedron.

The ElasticObject has the responsibility of caching nodal points for repeated execution of the same animation for use with the EA. For interaction the ElasticObject can create a Hold object with a set of nodes inside a given sphere.

As part of the evaluation of the surgical simulator with the surgeons, the following elastic models were implemented as described in the previous chapters:

- Linear static Finite Element method
- (Dynamic) Spring Mass system
- Quasi Static Spring Mass

## 6.2 Geometries

The main functionality of the Geometry class and specializations is to load and convert external file-format into the internal data structures. The Geometry class also defines a virtual method for cutting, given a cut-sweep, more about this in chapter 8.

The simplest specialization of the Geometry is the Surface class. The Surface class is a surface representation as discussed in section 5.9. The surface class can load a 3DS or OBJ file-formats. A set of nodes is created and the triangles are initialized over the set of nodes. Triangles thereby share nodes of the organ. A Triangle consists of three edges that are not shared with other triangles, i.e. the edges are local to a triangle [1]. Two edges that connect the same nodes reference each other through a mate relation. The mate structure is used to

[1] Even though several edges connect a set of nodes, only a single edge will be active in the simulation for use with e.g. the spring mass method.

make traversals along the surface, for use in the cutting procedure in chapter 8. Since a surface model has no tetrahedrons, none are defined.

The ConnectedSurfaces Geometry is a specialization of a general surface. The ConnectedSurfaces Geometry corresponds to the connected surfaces structure presented in section 5.9. The load operation of a ConnectedSurface still takes 3DS or OBJ formats, but can be used several times, adding surfaces to the model. After the last load operation, the surfaces will be connected to create the connected surfaces structure.

The TetrahedronMesh is a volumetric representation as discussed in section 5.9. The TetrahedronMesh can load a smesh [31] file-format containing a tetrahedron mesh. A Set of nodes is created on the basis of the information in the file-format. The set of tetrahedrons are initialized over the nodes, meaning that the Tetrahedrons share nodes. Each tetrahedron has four triangles, which are local to the tetrahedron. To allow for a traversal through the mesh structure from tetrahedron to tetrahedron, triangle mates are defined as the two triangles defined by the same three nodes (corresponding to the edge mate structure in the case of SurfaceGeometry).

# Chapter 7

# Interface: Interaction and Visualization

The previous chapters have defined the elastic behavior of tissue. To be able to actually use the surgical simulation as a tool for training or pre-operative simulation, we need an interface between the surgeon and the elastic model. The discussion concerning the interface in this chapter covers two areas: instruments for interaction and techniques for visualization.

## 7.1 Tools

In a surgical procedure many different instruments for a variety of uses exist. As part of a surgical simulator we would like to represent these instruments in the computer. We cannot simulate every part of the real surgical procedure, so the design of virtual instruments includes a recognition of what the defining properties of the instruments are. We would like the design to be general enough for it to be easy to extend with instruments that have properties in common with existing virtual instruments.

Bruyns and Montgomery dealt with this subject in the series of articles about virtual tools in the spring framework: [50, 10, 9]. "Virtual instruments" is a framework for a hierarchy of generalizations of real instruments [50]. Specifically the virtual instruments include an interface to different hardware components for interaction.

A general virtual instrument has a position and orientation in space. This information is transferred from a range of specific hardware components for interaction, through a specialization of a sensor class. The general virtual instrument offers methods for visualization and collision detection. These methods are used with the specialized geometry and collision response of a more specific tool.

The virtual instruments effectively reduce the phenomenon of a surgical instrument to the functionality of the instruments coupled with a geometrical

appearance. The different surgical tools are arranged in a hierarchy according to the functionality of the instruments; monolithic, hinged, telescoping and lasso instruments. In [9] virtual instruments for cutting are presented; scalpel, scissors and cauterizing wire. In [10] virtual instruments for probing, piercing and cauterizing are presented.

## 7.2 Collision detection

Collision detection in static geometry has been studied for a long time, and effective algorithms have been devised by partitioning the problem space into trees of geometrical components that can be searched quickly. The problem with collision detection in soft tissue is that the geometry is not static, and the acceleration structures will be invalidated quickly.

Different approaches have been proposed, such as a sphere tree collision detection scheme, with dynamic changes of the tree structure [7] and conservative sizes of leafs. In [28] the space is simply divided into a grid where collision detection is done by checking if voxels occupy the same grid point.

Another problem is that instruments often have a complicated geometry. In [9] it is recognized that only parts of the instrument's geometry are important for the functionality, and therefore also for the collision detection. E.g. only the sharp parts of a cutting instrument are relevant for collision detected.

The instruments are the primary sources of collision detection problems (if we exclude self collision). We can use the fact that the distance of the instrument in two consecutive frames is often not large. In [4] this structural coherency of movements of a scalpel is used to speed up the collision detection. When the scalpel has penetrated the tissue, the search for intersection until the scalpel leaves the tissue will use the structural coherency by first looking at the nearest neighbors, and then doing a full search. One could probably extend this scheme to a full breadth first search that would stop after a certain depth, after which a full search or an acceleration scheme should be used. An optimal depth could be found with respect to collision detection time.

## 7.3 Functionality of instruments

After a collision detection there is often a collision response. When an instrument has collided with some tissue, the constraint of non-inter-penetration should generate forces on the colliding objects.

The functionality of the collision response is instrument dependent; e.g. in [9] the collision response of a cutting instrument is discussed. If the instrument is moved in the direction of cut, the collision response will be a cut in the tissue. If the instrument is moved in a different direction, the collision response will be object deformation as in probing.

Cutting will be discussed further in chapter 8.

The main problem with grabbing and moving tissue is how it should influence the tissue. If nodes are grabbed, forces must be introduced (implicitly or explicitly) into the system to make the nodes follow the grab. With Spring Mass algorithms this reduces to methods based on forces added to the nodes or absolute positioning of the nodes. In [12] a penalty method is used to add forces to the nodes that interpenetrate some volume as follows:

$$F_c = \begin{cases} (-\lambda v - \mu \dot{v} v)k & \text{if } v < 0 \\ 0 & \text{otherwise} \end{cases}$$

$\lambda$ is the rigidity factor of the collision, $\mu$ is the damping factor, $v$ is the volume of the inter-penetration and $k$ is the contact direction. This is a heuristic set up to introduce forces that solve the inter-penetration.

Instead of virtual springs or forces such as the penalty based methods, one could argue that for rigid/soft deformations where the rigid part maintains its position, the nodes will end up in a specific location. We could just as well move the particle to that location. In [10, 73, 36] absolute positioning is used. The advantage is that we will get a realtime positioning of the nodes in question, the disadvantage is that these sudden, potentially large movements of nodes might induce instability into the system.

Absolute positioning of nodes is especially useful in connection with the Verlet integration because we have no explicit velocity variable that can grow out of sync with the position [36].

I have used absolute positioning of nodes in the surgical simulator implemented as part of this thesis.

## 7.4 Abstract tools

In the surgical simulator implemented as part of this thesis, the instruments only perform collision detection when they receive a use command (from a mouse click). The grab-tool will grab what is inside the sphere when used, and the cut-tool will cut what is penetrated by the cut-sweep when used. The collision detection used was therefore a simple exhaustive search.

If the tools implementation had not been an abstract implementation of instruments, and cutting should e.g. do progressive cutting while the cutting instrument penetrated the surface, then we would have needed a more complex collision detection.

The idea of having a generalized virtual tool that is specialized for special interaction or functionality has been used as the basis for the tools hierarchy in this thesis. The surgeons basically needed two categories of instruments: Grabbing and cutting. These instruments were implemented as abstract tools, the cutting instrument being a triangle representing the cut-sweep, and the grabbing instrument being a wireframe sphere representing the hand or instrument. The idea of using abstract tools instead of simulating the geometrical appearance was actually proposed by the surgeons. The skill of working with instruments does not depend on their visual appearance, and because the instruments are

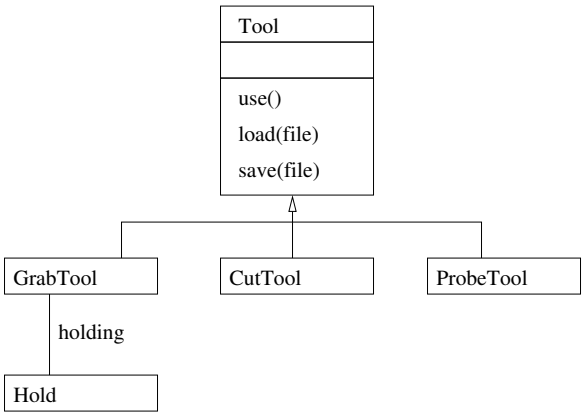



Figure 7.1: UML diagram of Interaction. The abstract tools are arranged in a hierarchy. The Tool class is a general tool implementation and implements the mouse control in 3d. A Tool can load or save interactions for use in the EA or for other comparisons. The use method represents a mouse click. The GrabTool gets a hold from the ElasticObject and manipulates the position of this hold. If there is a target ElasticObject for comparison of ElasticObjects the tool will also manipulate the position of the nodes in the target ElasticObject. The CutTool controls an abstract scalpel, which will issue a cutting operation on the geometry when the mouse is clicked. The Hold class implements a hold of a set of nodes, which can be moved around. The GrabTool gets a Hold from the ElasticObject within a given radius of a point

abstract they do not have to obey all the laws that normal instruments must. E.g. the abstract tools do not collide with the soft tissue (there is no collision response). This is an advantage because the target platform was a standard desktop computer with a mouse, and it can often be difficult to position the 3d point precisely in space with a mouse. This is an acknowledgment of the fact that we do not have to simulate every part of the real phenomenon, we can often get a better tool if we take into account the case of use, as explained in section 2.8.

If the main use case was different, abstract tools might not have been as effective. E.g. in certain training scenarios the tissue response to the handling of instruments is important as part of the exercise, because it can be difficult to control the instruments.

See figure 7.1 for an explanation of the abstract tools hierarchy.

## 7.5 Visualization

A range of different visualization techniques can be used, depending on the case of use. In training simulation realistic visual results can be very important. If the student is convinced that it looks real the entire experience might be more rewarding for him. Texture mapping is used within surgical simulation to add detail and make the object seems more realistic. A texture mapped organ will often add to the illusion of actual surgery. A 2d texture mapping has been used in [40] while a volume texture is used in [42].

Visual effects of bleeding, smoke, steam and irrigation can also be added to the simulation, to make the Surgical Simulation seem more real [40, 11, 2].

The visualization of the Surgical Simulator built as part of this thesis is based on OpenGL [55] rendering. Our primary case of use is pre-operative simulation; it is therefore important that we do not add incorrect information such as an artificial texture map to the organ. We have used simple flat or gouraud shading of triangles in the elastic model. Additionally visualizations specific to certain elastic models are possible, such as wireframe visualization of different spring categories. For comparison between two elastic models a visualization is supported in which the difference in nodal position is used as a vertex color.

# Chapter 8

# Altering Topology of Tissue

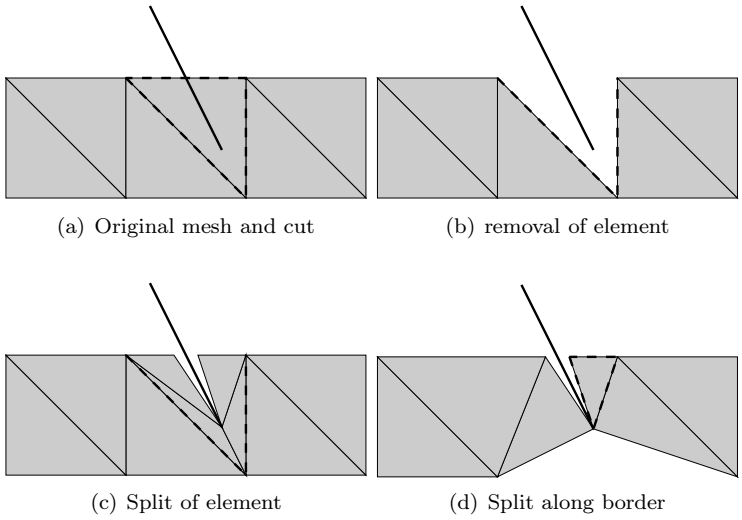
In section 2.2 on the VSD and section 2.3 on the ASD we saw that an important part of a surgical procedure is to make incisions into the tissue. If enough force is put upon tissue it may also tear. This chapter will deal with the issue of making topological changes, such as cutting or tearing.

Two important issues exist in enabling topological changes in realtime calculation of deformation. First of all the method used to calculate deformation must support changes in the topology fast enough for the simulation to still run in (near) realtime. The Spring Mass algorithm with the explicit integration schemes presented, explicitly supports topological changes because the model only uses local information in the calculation of the next step. We can simply add or remove edges and nodes and only makes changes in the connected elements. The Finite Element model as presented with an iterative solver of the linear system of equations also supports the change of topology. The methods using pre-calculation of inverse or factorization do not explicitly support topological changes. A change in topology will invalidate the pre-calculations and they will need to be recalculated, which may take a long time and destroy the illusion of realtime interaction, see section 4.5.2.

The second issue of cutting in the organ, is to make the actual changes in the connectivity of elements. Assuming that a cut is represented as some sweep through the organ, we would like to make changes to the organ in such a way, that the surface of the organ follows the sweep closely and does not intersect the sweep.

The important aspect of topological changes is topological complexity after a cut and how well formed the elements are after a cut. For a realtime application an increase in topological complexity is not wanted because it will increase the time to calculate deformation. The constraint of well formed elements is important because it might introduce visual artifacts as well as instability and incorrect data in the physical system.

Two different strategies exist for creation of the sweep. A static cut, where one indicates the total sweep of the cutting instrument - or a progressive method in which the organ topology changes every frame in response to the sweep [51].



(a) Original mesh and cut (b) removal of element

(c) Split of element (d) Split along border

Figure 8.1: In (a) the original mesh is shown. In (b) the cut is approximated to the removal of the elements inside the cut. In (c) elements are split into smaller elements to approximate the cut closely. In (d) elements are unglued from each other and moved to the cut

We will assume that the sweep is given as a number of triangles, based on which the entire cut is made.

In [9] it is recognized that the cutting algorithm must be defined in relation to the representation used to model the organ; surface, volumetric, multiple surfaces or hybrids.

## 8.1 Strategies for cutting

As stated in [61], cutting in tetrahedral or triangle meshes is difficult because tetrahedrons and triangles are not closed under cutting operations. A tetrahedron that is cut along a sweep does not simply result in another set of tetrahedrons. The mesh has to be adapted to make the cut appear where the user made the cut.

Basically three different models for cutting have been proposed as summarized by [22]. In Figure 8.1 the original mesh plus the three basic models of cutting are presented.

## 8.2 Removal of elements

The simplest cutting scheme is to simply remove the elements intersected by the cut-sweep [14, 28]. The problem with simply removing tetrahedrons from

the model is that the volume is not preserved. Furthermore the shape of the cut in the mesh will not resemble the actual cut, it will most likely be jagged and bigger than the actual sweep, because elements are most likely larger than the width of the cutting instrument.

The jagged cut can be smoothed out if the nodal points are translated to the cut sweep, this will be discussed in section 8.4.

No elements are added to the model, so the complexity is not increased, instead the complexity can decrease. This means we will lose detail when we cut into the model. This method is clearly most successful when the model consists of many volumetric elements, such as the 3d chainmail algorithm, see section 5.11.

## 8.3 Subdividing tetrahedrons

Instead of the very simple method of removing the elements, we could make a full decomposition of the tetrahedrons in the original topology to sub-tetrahedrons that divide the topology along the cut sweep. Elements in the triangles intersected by the cut and possibly neighbors are subdivided.

Such a method is presented in [4]. The advantage of subdivision of tetahedrons compared to simply removing the intersected elements is that the cut in the mesh is a very accurate representation of the actual cut. The volume of the organ is preserved, and no nodes are displaced.

The geometrical and visual result is very pleasing, but this has a price. The disadvantages are quite severe for our specific use of realtime interaction and visualization. A single tetrahedron intersected by the cut is replaced by a potentially large number of smaller tetrahedrons. This means that the basic number of primitives increases after a cut, and the speed of computation therefore decreases. The replaced tetrahedrons might be degenerate, that is, elements with a bad aspect ratio in which some edges are very long compared to others. The degeneracy can occur because the shape of the new tetrahedrons only depends on the original shape of the intersected tetrahedron, and the cut made by the scalpel. The mesh might become fragmented because it can become non-uniform with respect to the size of elements, which can have an influence on the numerical issues. Because of this fragmentation very small elements can occur, and if they are small enough they can have a negative influence on the stability of the numerical methods.

In a Finite Element system these degenerate elements will lead to numerical errors, specifically in the case of the Conjugate Gradient method, the convergence will slow down [21].

In the Spring Mass system degenerate elements will have a big influence on the behavior of the material as explained in section 5.10. An organ with degenerate elements might behave in a different way than if it was made from a regular grid of elements. Elements with small edges might also result in the error in which parts of the organ will collapse into itself.

In [22] an intersected element is divided into several tetrahedrons but a

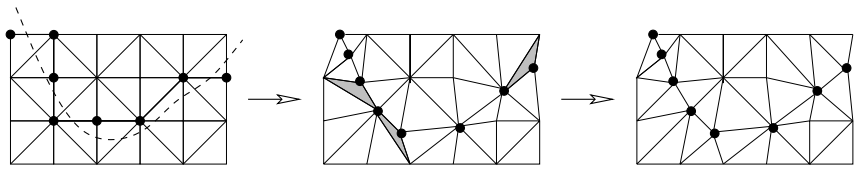


Figure 8.2: Splitting along faces; selecting faces to unglue, snap nodes to cut-sweep and remove degeneracy.

number of post-cut methods are devised to prevent fragmentation. The method is based on an edge-collapse scheme. The measure of quality of a tetrahedron is the ratio between the inscribed and circumscribed sphere times three (because the maximum of the ratio between the spheres is 1/3 and the ratio will then be between 1 and 0). The smallest edge is collapsed by moving one node into the other (and adding the mass). Although this method was not devised for the Finite Element method, modifications to the system can be made in realtime using some of the methods presented in this thesis.

## 8.4 Splitting along faces in the elements

We would like the number of primitives in the physical simulation to remain more or less constant. Consequently we will have to adapt the mesh without subdivisions to simulate cuts in realtime. In the series of articles by Nienhuys and Stappen [61, 59, 60] a scheme for cutting without subdivisions is presented. The cutting scheme presented has been implemented in the surgical simulator.

The approach for cutting is that the cut is approximated by a set of triangles in the mesh. The mesh is divided along the set of triangles found, and the nodes in the triangles are moved to the cut plane. The algorithm can be divided into several parts: Selecting faces to unglue, ungluing tetrahedrons that are connected to the faces, snapping nodes to the cut plane and removing degeneracy. The process is depict in figure 8.2

### 8.4.1 Selecting faces to unglue

The sweep is the surfaces indicated by the movement of the scalpel. The cut surface is the set of triangles from the tetrahedron mesh that most closely approximates the sweep. We will find a a cut surface from the cut sweep.

The cut can be approximated by a surface represented by triangles. In the case of our abstract cut tool, the cut sweep is a single triangle. The sweep is normally connected and non-branching, and we wish the cut surface to have the same characteristics.

The algorithm to find the cut surface from the cut sweep is presented in algorithm 5. The algorithm determines a feature set for each Tetrahedron, see figure 8.3. The feature set is the union of the nodes from each edge closest to

**Algorithm 5** find cut surface from sweep

```
for all (tetrahedrons t in mesh)
    E = edges of t that intersect the sweep
    for all(edges e in E)
        feature.add(node in e closest to sweep)
    if ( feature.size()==3 )
        cutSurface.add( feature
```

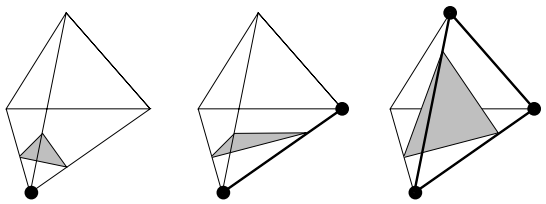


Figure 8.3: Feature selection of a point, an edge and a triangle respectively.

the sweep. If the size of the feature set is 1 a node is selected, if the size is 2 an edge is selected and if the size is 3 a triangle is selected. The feature set cannot be larger than 3 (see proof in [60]). We select feature sets of size 3 only, because we only need the triangles and features sets smaller than 3 are implicitly part of the triangle selected.

The algorithm 5 is not guaranteed to give a correct non-branching cut (see [60] for examples). In most cases the following will hold true:

- The cut surface is close to the sweep because only triangles from tetrahedrons that intersect the sweep are selected.
- The cut is most likely not branched, as at most one triangle is selected from each tetrahedron.
- The cut is most likely connected. Two adjacent tetrahedra will have the same nodes selected from the shared faces of the tetrahedra.

### 8.4.2 Ungluing faces

The outcome from the previous section is used to actually make the cut in the topology. That is, from a selection of triangles in the model, we would like to create a new model in which the boundary of the mesh is enlarged with the triangles representing the cut, and only those.

The unglue algorithm 6 takes a local look at each node and all the incident tetrahedrons. If the tetrahedrons are divided into distinct sets by the cut surface, the cut is realized by making a copy of each node in the cut-surface and replacing

**Algorithm 6** unglue in tetrahedral mesh

```
input(tetrahedra mesh,cut surface C)
for all( nodes N in mesh )
    T = the set of tetrahedra incident to N
    K = the set of components T is divided into by C
    for all(components c in K)
        Nc = N.copy()
        Substitute N with Nc in all tetrahedra of c
```

them in tetrahedrons on both sides of the cut. The algorithm is well defined in the manner that it creates valid tetrahedral mesh from a tetrahedral mesh. Most faces from the cut surface are put into the exterior of the mesh. The situation in which not all faces are put into the exterior is the situation in which a single triangle is selected. In order for the algorithm to behave as expected we need the initial mesh to be *cut-regular* that is:

**Cut-regular** A tetrahedral mesh is cut regular if for every node all the incident tetrahedra are connected through their faces.

If the mesh was not *cut-regular* the unglue algorithm could disconnect components of the mesh even if the cut surface was empty (see [61]).

### 8.4.3 Node snapping

Because the cut is made along the faces of the mesh, the resulting surface can be jagged. Node snapping is introduced to get rid of the jagged cuts that where introduced by cuts made along triangles in the model. The solution is to project nodes onto the cut sweep. This method can introduce unwanted deformations on the surface of the mesh because surface nodes are moved. One solution is to limit the movement of the node to within the surface triangle intersected by the sweep.

### 8.4.4 Degeneracy removal

Because nodes have been moved by node snapping, degeneracies can occur. In [61] a strategy for the different cases and solutions is listed. Degeneracie removal was outside the scope of the thesis and is not implemented in the surgical simulator.

## 8.5 Cutting in Connected Surfaces

The Connected Surface structure consists conceptually of two surfaces connected with additional edges. To make a cut through the Connected Surfaces structure, we need to cut in both the surfaces and the connecting edges.

**Algorithm 7** unglue in triangle mesh

```
input(triangle mesh,cut edges C)
for all( nodes N in mesh )
    T = the set of triangle incident to N
    K = the set of components T is divided into by C
    for all(components c in K)
        Nc = N.copy()
        Substitute N with Nc in all triangle of c
```

In section 8.4 we introduced a method to make cuts with minimal new element creation. The algorithm will be used as a basis for the cut in the surface structure. Other algorithms such as the Delaunay inspired approach by Nienhuys might give better results for triangle meshes, but are not easily generalized to tetrahedral meshes [63].

Selecting edges to unglue is done via feature selection of edges. This is the case when the feature set is of size two, see 8.3.

Algorithm 6 can simply be reduced to cutting in a triangle mesh, see algorithm 7.

For the Connected Surfaces structure it is not enough to simply cut in the surfaces, we also need to remove edges that intersect the cut sweep, as they connect the two sides of the cut. Due to the small number of connecting edges, the nodes in the cut can get underconstrained. This situation can be repaired by re-connecting the node to the opposite surface on the same side of the cut[1].

Cutting in Connected Surfaces is used in figure 8.4 where a cut is made into the heart. Notice that there is no visualization of a solid volumen between the surfaces.

[1]This is not done in the implementation though

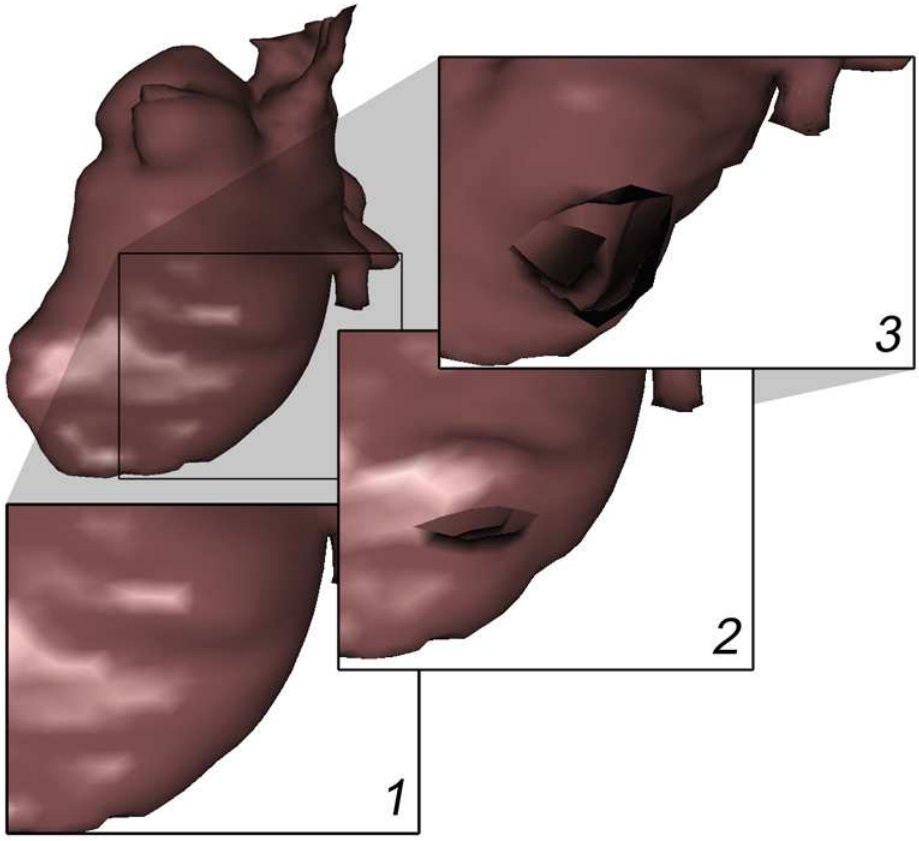



Figure 8.4: Cutting in heart

# Part III

# Validation

# Chapter 9

# Parameter Optimization

In this chapter we will compare the actual behavior of the models introduced in this thesis. Specifically we will compare the the Spring Mass models with the FEM to see to what degree this family of elastic models can approximate the behavior of FEM and what framerate and convergence issues there are.

To make comparisons that actually compare the models on an equal basis, we will need optimal parameters for the Spring Mass models with respect to the behavior of the FEM. As discussed in section 5.10, we cannot simply transfer the parameters from the FEM to the Spring Mass model.

In [33] two dynamic elastic models are compared. The reference is a simplified time explicit FEM and the elastic model under study is a linear Spring Mass model. The parameters for the models are manually tuned to make the Spring Mass model behave as the FEM. A force is introduced into the two models and the time it takes for the models to reach equilibrium is found. There are several shortcomings in such a test. The material parameters are manually tuned. This is unsatisfactory because it is the behavior of the elastic models that is examined. The models should be compared on an equal basis with optimal parameters for comparison - we have no guarantee with manually tuned parameters. The time it takes for the model to reach equilibrium is measured, but in real use of a surgical simulation the equilibrium is not reached before new forces are introduced through interaction with the model. Furthermore equilibrium in itself does not tell us if the models have reached the same equilibrium; this depends on the material parameters. As a side note, it seems odd to compare two dynamic models for static equilibrium; there is no comparison of the dynamic properties of the materials.

In [20] a Simulated Annealing method is used to optimize the elastic behavior of a dynamic Spring Mass model to a dynamic FEM. The test cases used are four basic deformations on a regular square plate in 2d, two stretching and two shearing. The reference displacement is an analytical calculation of the deformation of a FEM. The fitness of the model is calculated as the standard deviation. A Simulated Annealing is used because a pure gradient-descent delivered bad results (probably because of the dimensionality of the problem and local min-

**Algorithm 8** Standard EA

```
while (stopcriterion)
    parents = population.selectParents()
    nextGeneration = parents.recombine()
    nextGeneration.mutate()
    nextGeneration.addFitSurvivers(population)
    population = nextGeneration
```

ima). The optimization presented only optimizes spring stiffness. Stepsize and damping are not optimized. Again, a comparison for equilibrium is used in the fitness analysis. The convergence (which can often not be completed in a single frame) is not analyzed.

I will find an optimal parameterization of an elastic models such that it will behave as closely as possible to a static solution to a linear FEM. The FEM used as a reference is an actual solution to the linear system of equations, and does as such not run in realtime. The important contribution in my parameter identification, is that I optimize for a behavior over time. My parameters will be optimized for a reference model running at a fixed speed. The target model runs at whatever speed that specific model supports. For each frame of the reference model, the two models are compared and a fitness measure is calculated.

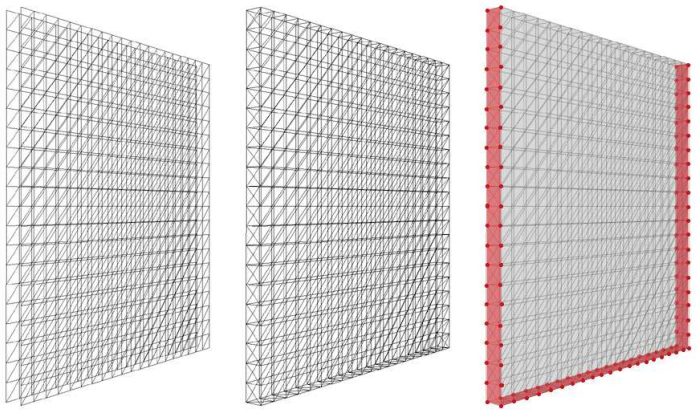
I have optimized for a specific behavior as experienced through the simulator. Initial experiments showed me that it was not realistic to hope for equilibrium in each frame, neither with quasi static or conjugate gradient FEM. The fitness of a given parameterization is a measure of how good the model exhibited the same behavior in the same period of time.

In [33] hexahedrons are used as elements in the FEM formulation - in order to convert these to a Spring Mass system that is not under-constrained they need to insert diagonal springs. In my case tetrahedrons have been used as elements in the FEM formulation, and no extra springs need to be inserted.

## 9.1 EA for parameter optimization

The implemented system for optimizing parameters supports a comparison of two elastic models through the ElasticComparison class. There is a target elastic model and a reference elastic model. The idea being that we compare the target model to a more precise reference model.

We use an Evolutionary Algorithm (EA) [46, 47] for our optimization problem. Not knowing anything about the behavior of our fitness I select the EA because it can escape local optima and only needs a fitness evaluation of a parameterization to make a search for optimal parameters. An EA is inspired by the classic theory of Darwinistic evolution; The fittest individuals survive, mates with other fit individuals and transfer parts of their genetic code to their offsprings. In the EA a population converges to the global optimum through a



(a) ConnectedSurfaces wall (b) TetrahedronMesh wall (c) Fixed nodes in red

Figure 9.1: The test geometry: A thin wall. (884 nodes, 2408 tetrahedrons and 9766 edges)

series of generations.

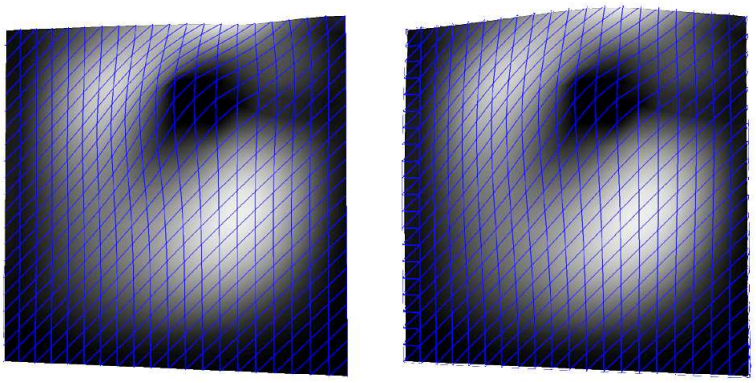
The fitness function $fitness(p_1, p_2, ..., p_n) \rightarrow [0, 3]$, where $n$ is the number of parameters we wish to optimize, defines a *fitness landscape* which I will visualize in the cases possible and use to evaluate the behavior of the elastic model.

A basic EA is presented in Algorithm 8. From the current generation we select parents to be recombined or transfered to the next. Parents are selected based on their fitness. A fit parent is one that behaves like the reference model. The next generation is mutated slightly. The selection used is tournament selection because we expect the fitness landscapes to be rather smooth, and convergence should not be a problem. When we recombine, we create a new individual whose parameter values are the average of it's parent's numerical values. Mutation adds or subtracts small numbers from the parameters. The most fit individual from the previous generation will be transferred to the next. The EA used in the optimizations has a population size of 20-45, crossover rate of 0.5 and a mutation rate of 0.5. The initial chromosomes are scattered randomly in a part of the search space.

### 9.1.1 Comparing models

We now define how to compare two models and calculate fitness. We must define what it means for two models to behave alike over a time period.

We compare the models in an experimental setup, meaning that they are run under some test interaction and compared. Three things must be defined for such a test: the metric needed to compare two solutions, the test interactions that are to be performed, and the geometrical object on which the test is run.



(a) Connected surfaces deformation (b) FEM deformation

Figure 9.2: Nodal fitness mapped onto geometry. Deformation as in figure 9.3.

The Geometrical Object used in the comparison is a thin wall. In figure 9.1 the wall is depicted as two surfaces (a) and the tetrahedron mesh (b) . The wall model is chosen because the Connected Surfaces model is designed specifically for this type of model because of the heart morphology, see section 5.9.4. The wall is constructed as a well formed 3d object with equal spacing between nodal points. The wall is fixed at three of the three sides of the wall. In [20, 33] a brick is used as the test case.

The test interaction is a simple grab and stretch. The stretch movement lasts for 1.5 seconds follow by 3 seconds with no movement allowing the elastic models to find equilibrium. The deformed wall can be seen in figure 9.3.

The metric is defined as the average of the distance between nodes paired in the two models. The nodes are paired through their position in space. Through the geometrical Object setup we are guaranteed that all points in the target model are present in the reference model. The fitness at a single timestep is calculated as:

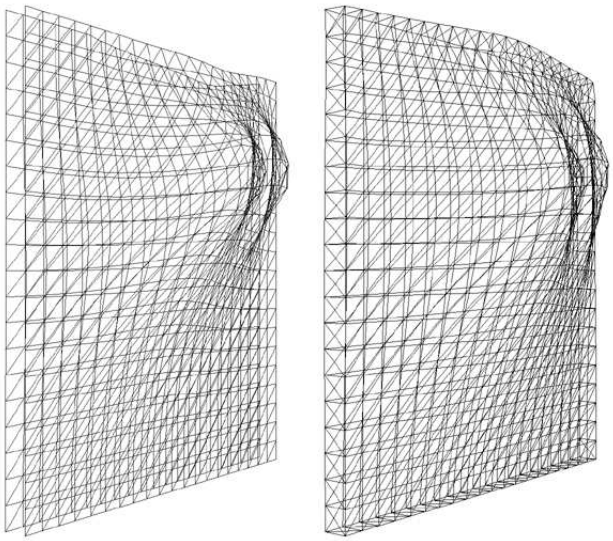
$$fit = \sum_{i=1}^{N} \left\| x_i^t - x_i^r \right\| \frac{1}{N}$$

The difference between the reference model and the target model on a nodal basis can be visualized in the Surgical Simulator[1]. In figure 9.2 a comparison between a Connected Surfaces model and a FEM is made.

The target elastic model runs at whatever speed possible, while the reference model runs at a fixed speed (the nodal positions being pre-calculated).

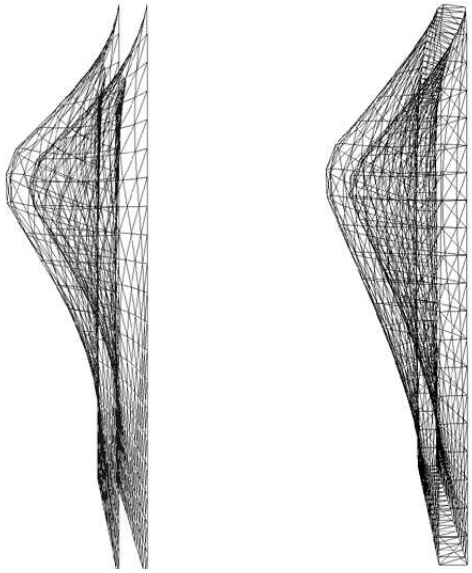
Because the same reference model is compared many times to the target elastic model for which we are finding parameters, the animation of the reference

[1]Press '6' to activate this view in the surgical simulator.



(a) Connected surfaces deformation

(b) FEM deformation



(c) Connected surfaces deformation from side

(d) FEM deformation from side

Figure 9.3: Test Interaction

**Algorithm 9** Evaluate fitness

```
INPUT(target,reference)
OUTPUT(Fitness)
while( tool interaction is not finished)
    tool.timeStep()
    reference.cachedTimeStep()
    for ( 1/30 of a second )
        target.setHoldPositionFrom(reference)
        target.timeStep()
    difference += Compare(target,reference)
fitness = difference / tool.totalTimeSteps()
```

model is cached. This is especially useful for the precise FEM because it is computationally heavy.

The complete algorithm to evaluate the fitness is presented in Algorithm 9.

The problem of finding both parameters for the speed of convergence (step-size and damping) and the precision of the equilibrium is actually a multi-objective optimization. As explained in section 5.3, the precision of equilibrium and speed of convergence depend on each other. If we want a precise calculation of equilibrium, we must use small time steps - and if we use large time steps the equilibrium is not precise. The question is how these two parameters behave with respect to the FEM. The tradeoff between the two parameters in the fitness evaluation is implicit in the interaction test, and depends on the ratio of the time the tissue is interacted with and the time it is not. The size of the forces also influence the two parameters.

It is important to recognize the fact that we are not trying to find parameters that will make the elastic models run faster. The speed of the elastic models is part of the definition of the elastic models.

### 9.1.2 Chromosomes

The stiffness of the springs are generally homogeneous, i.e. all springs have the same stiffness parameters. We could have chosen to find the optimal stiffness parameters for each individual spring. But as the reference model is homogeneous, optimal individual stiffness parameters would probably be adapted to the interaction case used. That is, we would find a parameter configuration that would work well with the interaction cases chosen, but not in the general case (as the material would not behave in a homogeneous manner in all cases). Furthermore individual spring stiffnesses would increase the dimensionality of the problem to an amount not easily solvable with a general EA.

## 9.2 Minimum and Maximum fitness

To interpret the fitness calculations, we will do a small test to see what the worst realistic result is. We will measure the fitness of a FEM compared to an elastic model in which only the points in the Hold object are moved. We move the points in the Hold because these will move correctly in all cases because the positions are merely transfered. The fitness in such a situation has been experimentally established to be 0.202279 and will be regarded as a maximum fitness.

As an indication of a minimum fitness, a static equilibrium in each frame is calculated with the quasi static algorithm. A static equilibrium with the quasi static algorithm in each frame is not possible in realtime though. If equilibrium is found each timestep, the fitness becomes 0.0482.

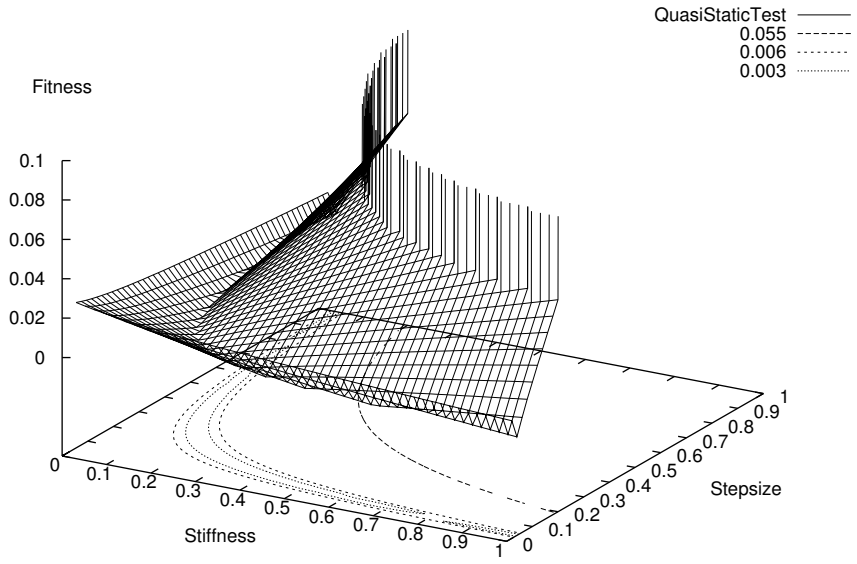
## 9.3 Tetrahedral QuasiStatic to QuasiStatic optimization

In this preliminary test we compare a quasi static model to a quasi static model. This optimization is merely to verify that the EA will find an absolute minimum fitness of zero for a simple case. The reference model was set to have a minimum in $(stepsize, stiffness) = (0.2, 0.2)$. The fitness found was practically zero, but the parameters were not quite $(0.2, 0.2)$ though. See the table 9.1 and the convergence in figure 9.5. The answer to this issue is found by inspecting the fitness landscape in figure 9.4, where we can see that there is a valley in which the minimum value is reached. We can see that stiffness and stepsize depend on each other. The stepsize and stiffness are interchangeable in the QuasiStatic model when we use absolute positioning of the nodes. The intuitive explanation is very simple; the stiffness indicates how far the direction of velocity should be followed and so does stepsize. This is especially true in a homogeneous setting in which all spring coefficients are the same. Note that this is only true because we work with absolute positioning of the interaction points. If we were using forces for interaction, a model with greater spring stiffness would resist more to the force.

| Stepsize | Stiffness | Fitness |
|---|---|---|
| 0.218276 | 0.18789 | 0 |

Table 9.1: Minimum stiffness of Tetrahedral QuasiStatic to QuasiStatic optimization.

From figure 9.4 we can also see that the minimum is far away from the instability border - and though instability can be introduced through large forces, there is good distance to the instability border. In this first test, the two models were run at the same speed, to be sure that there would be a minimum in




(a) 40x40 grid of fittness landscape

Figure 9.4: Fitness landscape of optimization of QuastiStatic parameters to a QuasiStatic model. Contours at 0.003, 0.006 and 0.55 are shown to indicate fitness values in the fitness landscape and ease the interpretation of the 3d graph on paper

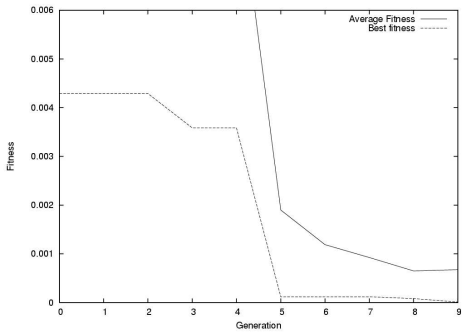



Figure 9.5: Convergence of the EA for an optimization of QuasiStatic parameters to QuasiStatic model.

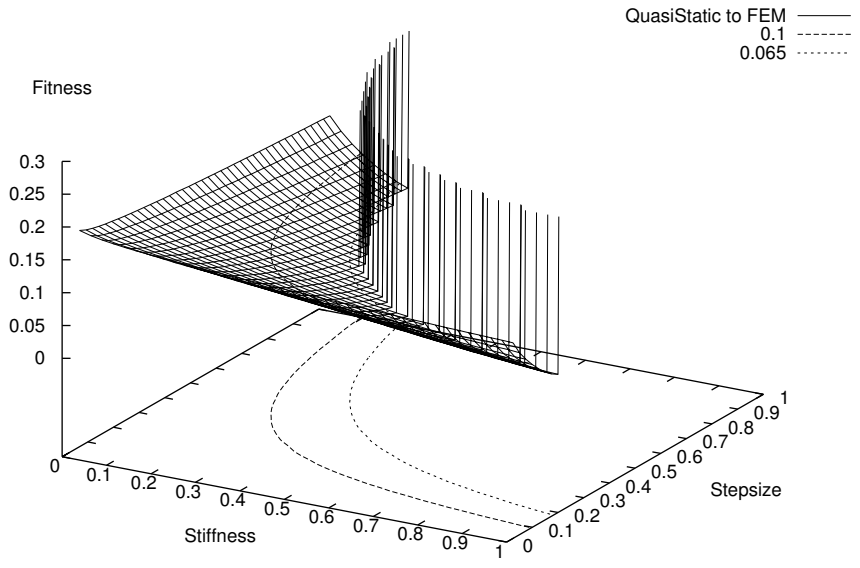



Figure 9.6: Fitness landscape of optimization of parameters for a QuasiStatic model to FEM (40x40 grid)

$(0.2, 0.2)$.

## 9.4 Tetrahedral QuasiStatic to precise FEM optimization

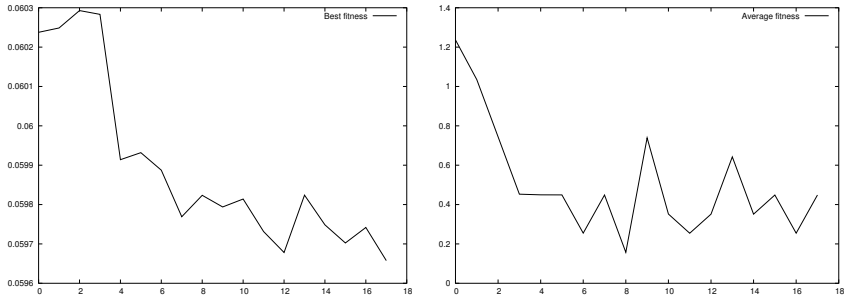
For the Tetrahedronal QuasiStatic model we wish to find to the optimal stepsize and stiffness parameters. The fitness landscape is 2d, depicted in figure 9.6.

The actual minimum found is presented in table 9.2.

| Stepsize | Stiffness | Fitness |
|---|---|---|
| 0.20286 | 0.490712 | 0.05905 |

Table 9.2: Minimum fitness of optimization of parameters for a QuasiStaic model to a FEM

Compared to the previous section, we can see that the valley in which a minimum is reached, has shifted into the wall of instability. This indicates that




(a) BestFitness (b) Average Fitness

Figure 9.7: Convergence of the EA for an optimization of parameters for a QuasiStatic model to a FEM

the Spring Mass system is having difficulties behaving stiffly enough or converging quickly enough. The optimal parameters will be very close to the border of in-stability, and other interactions with a model based on these parameters might become unstable.

The problem with the instability can be seen in Average fitness and Best fitness in figure 9.7 and figure 9.6. In the Best fitness figure, we can see that the convergence of the EA becomes quite jittery after generation 7. This effect appears because small differences in the duration of fitness evaluation can mean that a parameterization that was stable in the previous generation becomes unstable in the current. In the case of the best fitness another parameterization will become the best, but possible with a higher fitness. In the Average Fitness figure we see the same problem as large oscillations after generation 7. An unstable parameterization is punished with a large fitness adding a large amount to the average fitness.

The problem is simply that the QuasiStatic Spring Mass model cannot converge quickly enough. One idea is to use relaxation to increase stability and speed of convergence.

## 9.5 Tetrahedral Relaxation

If relaxation (from section 4) is the only deformation calculation we make, we get a result a bit worse than the pure Spring Mass model deformation, see table 9.3. This is not really a surprise as relaxation in itself is only a heuristic deformation. But it tells us that relaxation in itself will not give us good fitness.

| Linear factor | Relaxation iterations | Fitness |
|---|---|---|
| 0.000351211 | 1 | 0.0622297 |

Table 9.3: Minimum fitness for a parameter optimization for a tetrahedral relaxation (in the range 0 to 3) model compared to a FEM

## 9.6 Tetrahedral QuasiStatic with Relaxation

The QuasiStatic Spring Mass algorithm has a problem behaving stiffly enough and propagating forces quickly enough. We will therefore try to use the iterative relaxation introduced in section 4. We hope that relaxation will propagate forces more quickly.

The problem has increased to a four dimensional problem of stepsize, stiffness, number of relaxation iterations, and the linear factor of springs.

To get information about the behavior of the QuasiStatic algorithm with relaxation we would also like to visualize the fitness landscape. Unfortunately this is a 5 dimensional dataset. From the figure 9.8 (a) we can see that the linear factor of the relaxation is best left close zero[2]. Furthermore we have experienced some dependency between stepsize and stiffness, meaning that we can decide on a certain stepsize, and only visualize the stiffness. We then have a 3 dimensional system of number of relaxations steps, stiffness, and fitness visualized in figure 9.8 (b).

The fitness landscape in figure 9.8 (b) is not as smooth as the previous fitness landscapes simply because of the relaxation. It is not immediately clear where a minimum could be.

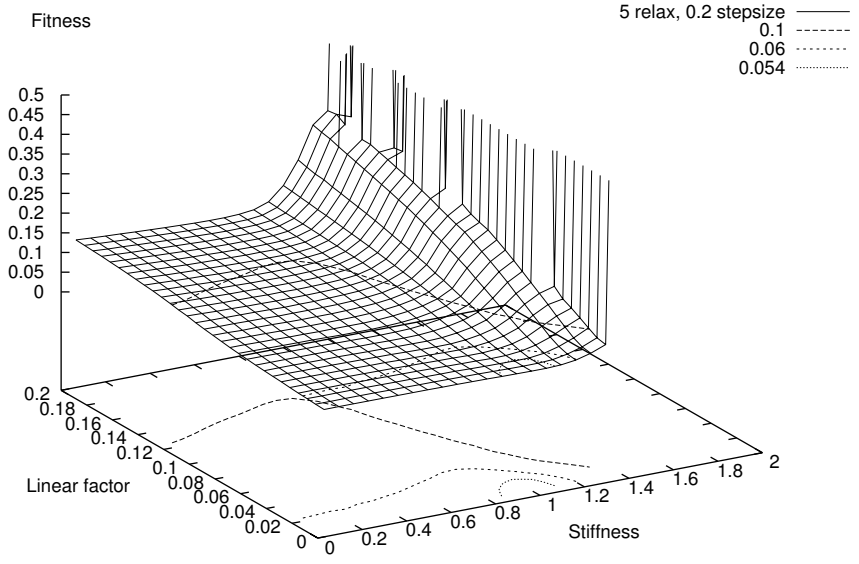
The convergence in figure 9.9 is as expected. Comparing it to the convergence of the Quasi Static algorithm without relaxation, we get some indication that the minimum is more stable because of fewer and smaller oscillations. From the best fitness we can clearly see that the optimal solution is probably not right next to an instable one, because the graph is smooth all the way.

| Stepsize | Stiffness | Linear factor | Relaxation iterations | Fitness |
|---|---|---|---|---|
| 0.0542248 | 0.446545 | 0.00507468 | 10 | 0.0497 |

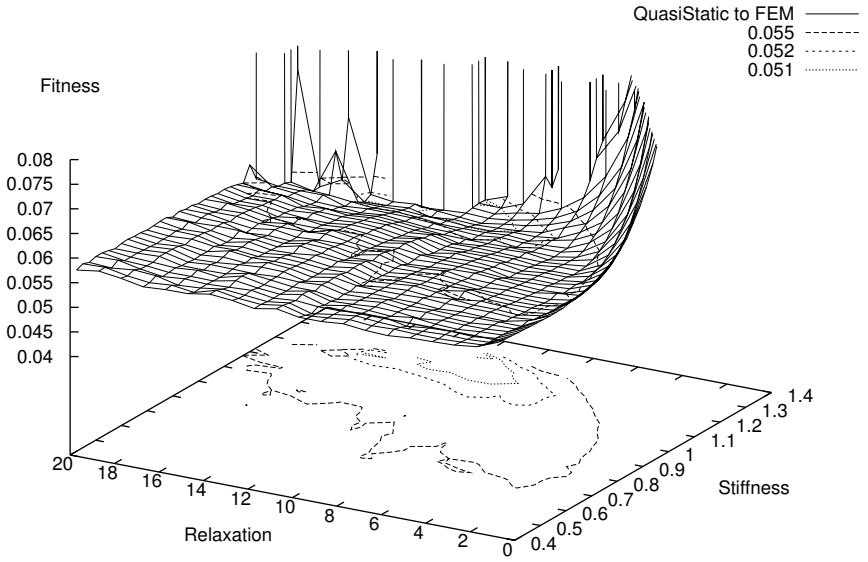
Table 9.4: Minimum fitness for a parameter optimization for a tetrahedral QuasiStatic model with relaxation (in the range 0 to 15) compared to a FEM

We initially seeded the EA with a number of relaxations up to 15, this gave a minimum of 11 iterations and a fitness that is 0.01 lower than without relaxation, see table 9.4. This is a relatively good result, but at a price. 11 relaxations are large number of relaxations giving an unacceptable framerate.

[2]Although a visualization is only made for 5 relaxations, similar results occur with larger or smaller number of relaxations.




(a) Test of Linear Factor (5 in relaxation, 0.2 in stepsize)




(b) Relaxation fitnesslandscape (0.2 in stepsize, near 0 LinearFactor)

Figure 9.8: Fitness landscape for an optimization of parameters for a Tetrahedral QuasiStatic with Relaxation to a FEM

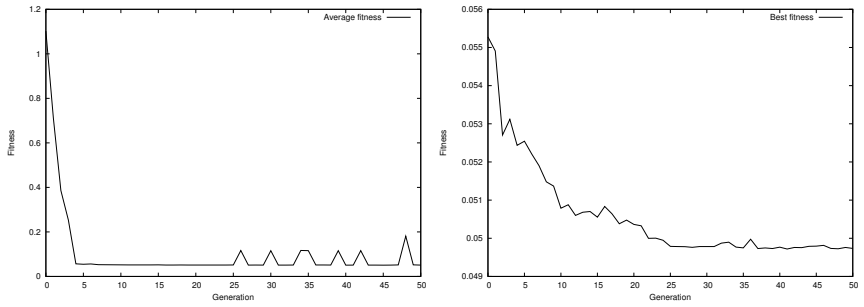



Figure 9.9: Convergence of an EA optimizing parameters for a tetrahedral QuasiStatic model with relaxation to a FEM

The problem is that there is nothing in the optimization that optimizes for frame rate. If we constrain the number of relaxations to a smaller number such as three relaxations per frame, the fitness is similar, see table 9.5. In all cases, the fitness is better then both pure Spring Mass and pure Relaxation deformation, indicating a synergy effect.

| Stepsize | Stiffness | Linear factor | Relaxation iterations | Fitness |
|---|---|---|---|---|
| 0.31525 | 0.490227 | 0.0235962 | 1 | 0.0494 |

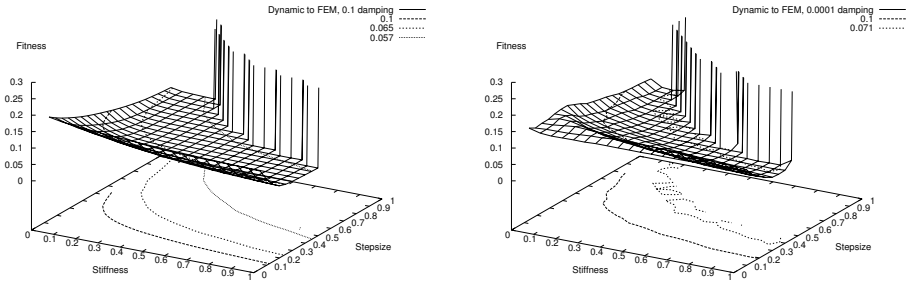
Table 9.5: Minimum fitness for a parameter optimization for a tetrahedral QuasiStatic model with relaxation (in the range 0 to 3) compared to a FEM

## 9.7 Dynamic Spring Mass

This section deals with a tetrahedral mesh controlled by a dynamic spring mass algorithm. We therefore have a 3d problem: stepsize, stiffness and damping.

The dynamic Spring Mass system exhibits a time dependent movement, with waves and vibrations. The reference FEM is static, and such phenomena do therefore not appear. In figure 9.10 we compare a low degree of damping (b) and a high degree of damping (a). We see that when the damping gets higher the model receives a lower fitness because waves and vibrations are damped out. On the other hand, if the damping gets too big, the convergence is slowed down. The Dynamic Spring Mass model must therefore optimize its parameters for a static behavior.

Surprisingly the dynamic system is able to get a lower fitness than the quasi static. It seems the velocity of the nodes might indicate a search direction for the nodes, thereby accelerating the convergence. Another source of the better fitness could be that the integration used in the dynamic calculation is more advanced than the simple method used in the QuasiStatic model.




(a) high damping

(b) low damping

Figure 9.10: Fitness landscape of the optimization of parameters for a dynamic Spring Mass model in comparison to a FEM

| Stepsize | Stiffness | damping | Fitness |
|---|---|---|---|
| 0.589101 | 0.740875 | 0.0665089 | 0.0555 |

Table 9.6: Minimum fitness for an optimization of parameters for a dynamic Spring Mass model in comparison to a FEM.

What we cannot see from the optimal parameters, is how the dynamic spring mass model achieves such a low fitness. It might simply optimize for features in the test interaction. We will compare the models for speed of convergence in section 9.9.

## 9.8 Quasti Static Connected Surfaces

In this section we will examine the Connected Surface structure from section 5.9.4. The Connected Surface divides the springs into two categories; surface springs and connecting springs. As a first test we will optimize for a homogeneous springs constant. The result is presented in table 9.7. We can see that the optimum becomes 0.006 worse in comparison to the homogeneous tetrahedral mesh.

| Stepsize | Stiffness on surface | Stiffness in connecting springs | Fitness |
|---|---|---|---|
| 0.967284 | 0.402214 | 0.402214 | 0.0661307 |

Table 9.7: Minimum fitness for a ConnectedSurface Spring Mass model in comparison to a FEM. Stiffness on surface and in connecting springs are the same.

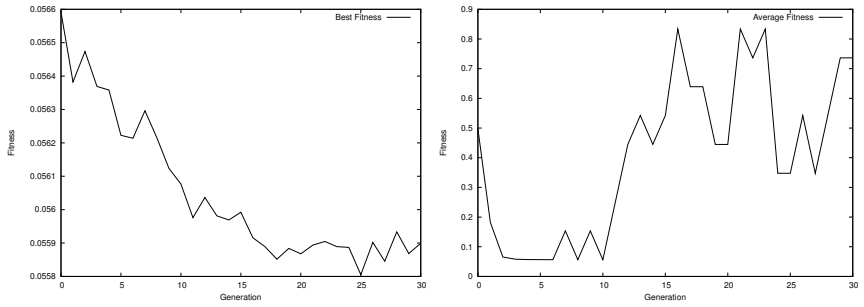



Figure 9.11: Convergence of an EA optimizing parameters for a tetrahedral dynamic Spring Mass model to a FEM

The two categories of springs in Connected Surfaces serve different needs in the calculation of deformation, see section 5.9.4. In table 9.8, the results for a separate spring stiffness for the springs on the surface and in the volume is presented. We notice that the fitness gets better, but is still not as good as in the tetrahedral mesh case. The difference in fitness is probably because the scheme of connecting surfaces is not optimal for our test case in which direction of the connected springs points directly into a node in the other surface. With the low number of connecting springs, the propagation of force can easily become biased in certain directions.

| Stepsize | Stiffness on surface | Stiffness in connecting springs | Fitness |
|---|---|---|---|
| 1.10633 | 0.197968 | 0.755308 | 0.0639859 |

Table 9.8: Minimum fitness for a ConnectedSurface Spring Mass model in comparison to a FEM

The convergence of the EA optimizing the Connected Surfaces case (see figure 9.12) is very slow in the in comparison to the other optimizations - this indicates that the interplay between connecting springs and surface springs is not easily recognized.

## 9.9 Comparison of models

In [57] Spring mass is used in the simulation of abdominal simulation with cutting because it is proclaimed that Spring Mass algorithms are faster than FEM when cutting should be possible. The conclusion in [33] is that dynamic FEM and Spring mass have the same order of computational complexity.

In [38] a comparison of spring mass and FEM is made within the domain of craniofacial surgery. It is informally verified that spring models behave similarly

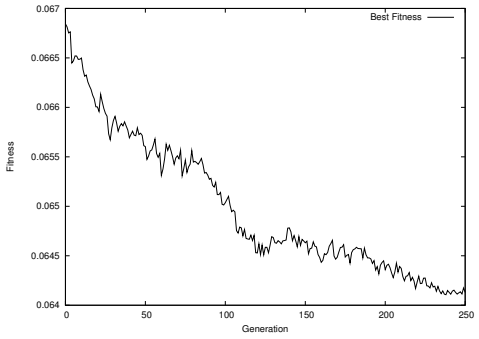



Figure 9.12: Convergence of an EA optimizing parameters for a ConnectedSurfaces QuasiStatic Spring Mass model to a FEM

to FEM for small forces. The Spring mass system is used as a fast realtime simulation, and the FEM can be used to make off-line calculations of the same procedure to verify results. For larger forces the behavior of the models are not identical.

Using the optimal values for the elastic models studied, we will now study the convergence behavior of the elastic models. We will use the same interaction sequence as we used to find the optimal parameters.

The graphs can be seen in figure 9.13. At 0 seconds the stretch is started and at about 1.5 seconds it is stopped. We can see that none of the elastic models can keep up with pre-calculated FEM; while there is a stretching all the graphs are rising. The jitter on the first parts of the graphs comes mostly from the interaction sequence, which is not smooth (as it is recorded in a real setting). Part of the jitter also comes from the iterative algorithms.

Comparing the Conjugate Gradient FEM to the Spring Mass models, we can see that the CG FEM does not mimic the reference model as closely as the Spring Mass models for the first 4-5 seconds. But because the CG FEM and the reference model are actually the same, the CG FEM will converge to an actual 0 in fitness after about 18 seconds, see figure 9.13 (b). The choice of CG FEM and Spring Mass model depends on the choice between a fast reasonable result and a slow precise result. For the use of a realtime interaction, fast response is important for the credibility of the behavior of the tissue. The Spring Mass model seems better suited for this kind of use. Another argument is that slow convergence gives the impression of a material that is too elastic to be tissue.

In the part of the graph where there is an active stretching, the static tetrahedral Spring Mass model has the worst fitness. The ConnectedSurfaces Spring Mass model looks almost the same, but is a bit better. The Relaxation of the static tetrahedral Spring Mass has a lower fitness again, but the dynamic tetrahedral Spring Mass has a very low fitness on the first part of the graph.

The Spring Mass models can acquire a faster convergence in the first seconds,

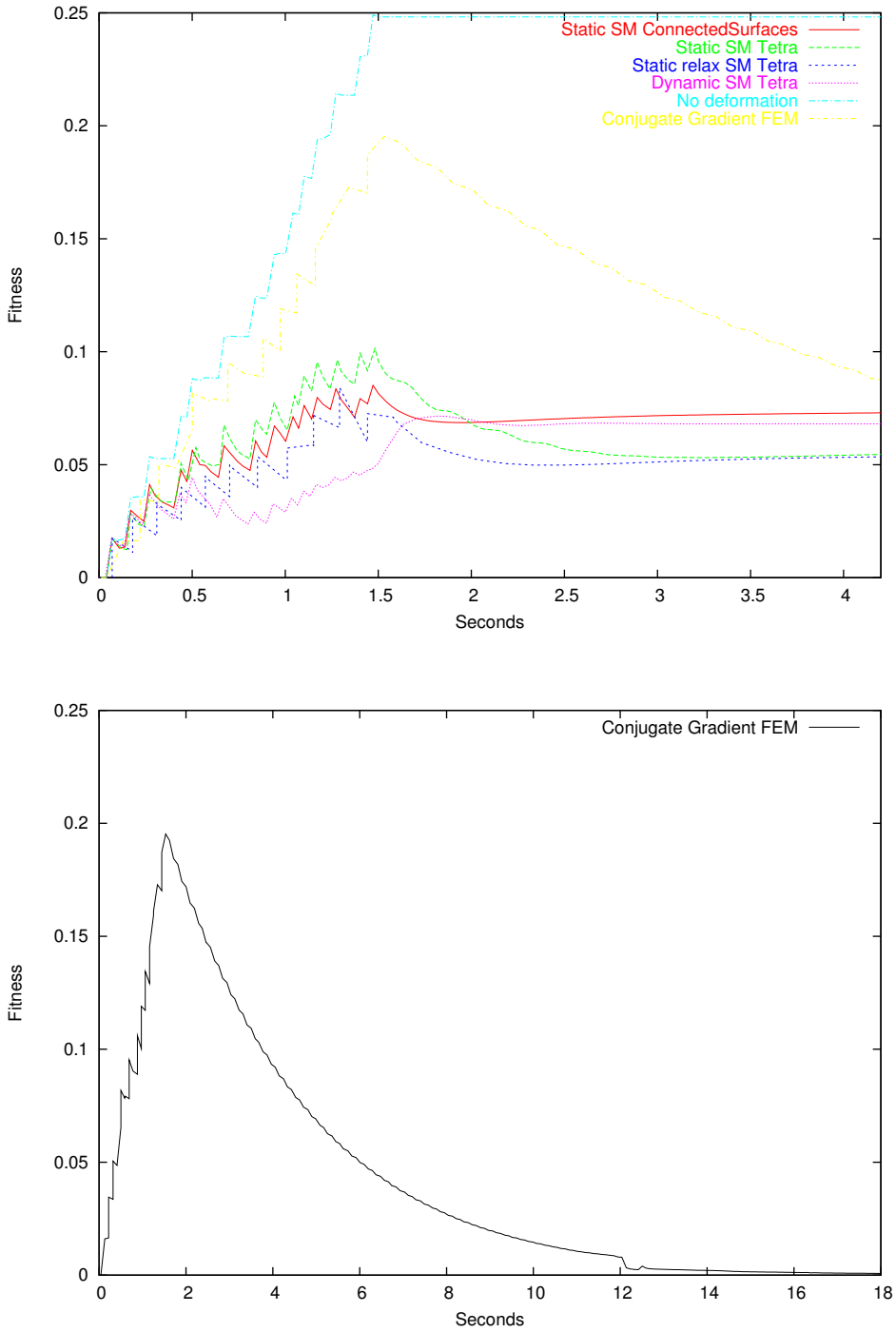



Figure 9.13: Convergence of the elastic models with optimal parameters in comparison to the behavior of a FEM.

| Elastic Model | Frames pr second |
|---|---|
| QuasiStatic Tetrahedral | 57 fps.[3] |
| QuasiStatic Connected Surfaces | 90 fps. |
| QuasiStatic Connected Surfaces, one relaxation pr frame | 61 fps. |
| one relaxation pr. frame | 106 fps. |
| Dynamic Spring Mass | 86 fps. |
| CG FEM | 12 fps. |

Table 9.9: Framerate of elastic models with the wall model

but because they are unstable and the best convergence is close to instability, the optimal parameters are often unstable. To find stable parameters we should optimize through a large number of interactions in a safe range around the force of interaction used in an actual training scenario or pre-operative simulation.

From table 9.9 we can see that the Spring Mass models and pure relaxation run at about the same speed of about 90 frames per seconds.

If we introduce a single iteration of relaxation pr. frame, we drop about 30 frames pr. second. In the case of the LR Spring Mass the drop in framerate is acceptable because it enables us to calculate the Spring Mass response in a smaller area, thereby speeding up the framerate. Furthermore relaxation helps the convergence in large models.

The CG FEM performance is unacceptable for realtime use, 12 frames pr. second is not fast enough for realtime use. The heart geometry is even bigger, and would give too slow framerate for the surgical simulator to be useful as a realtime tool. CG FEM has not previously been compared in detail to the Spring Mass model. [61] simply states that realtime performance is possible in their implementation. I have opted for the same level of optimization in the code, and it seems that CG FEM is inferior to the Spring Mass model when it comes to initial fast convergence and framerate. CG FEM on the other hand delivers a precise equilibrium with global behavior.

To sum up, the Spring Mass model gives a better framerate and faster initial convergence than a CG FEM. The fast initial convergence that stagnates is preferable to a slow convergence that reaches the "correct" minimum because we are concerned with realtime use. If the Spring Mass model is combined with relaxation, we get a better convergence - especially for large models such as the heart geometry. The LR Spring Mass model uses these fact to simulate the deformation in a large heart geometry at interactive rates with a fast convergence to equilibrium.

[3]Due to implementation issues, all springs are active (in all tetrahedrons) and therefore restricts the framerate. This has no influence on the parameter optimization though

# Chapter 10

# Evaluation with Surgeons

Building a useful surgical simulator demands cooperation and evaluation with surgeons. The expert evaluation is proposed in [27] as one kind of validation. The design of the surgical simulator was presented in chapter 6. The surgical simulator was evaluated in cooperation with Ole Kromann Hansen and Vibeke Hjortdal. The surgeons were generally very positive towards the surgical simulator, see appendix D

The actual evaluation with the surgeons was an iterative process in which elastic and geometrical models were presented and discussed. Different geometries were used in the evaluation, ranging from simple walls and spheres to the full cardiac geometry.

The surgeons were exposed to most of the models and alternatives discussed in this thesis. The iterative evaluation with the surgeons has resultet in the LR Spring Mass model. This model is designed specifically for the cardiac geometry (the wall perspective) and realtime calculation through point interaction, regions-of-interest and fast convergence. LR Spring Mass model (section 5.7) was found to be very realistic, and the surgeons could easily see how such a tool could be useful as a pre-operative tool. Interaction is based on abstract tools (section 7.4) with absolut positioning of nodes.

The evaluation of elastic models and geometry is conceptually divided into six subjects. The elastic behavior is divided into *deformation equilibrium*, the choice of *dynamic or static model* and *deformation animation* ( such as time dependant movement, speed of convergence etc.). The general use or the simulation techniques are evaluated through subjects; *interaction*, *simplifications* of reality and *categories of use* for the surgeons. I will furthermore reflect on the models in comparison to the basic observation in section 2.6.

## 10.1 Deformation equilibrium

In the basic case of a Spring Mass model and the FEM, the deformation equilibrium is determined by the material parameters and the choice of the linear

stress-strain model. The linear equilibrium was found to be realistic in most cases. The surgeons noticed that the deformations were unnaturally elongated when they introduced large forces into the surgical simulator. The size of the forces introduced were far beyond a realistic stress on the cardiac tissue of a child, but because of the choice of a mouse as an input device there are no constraints or haptic feedback to guide the amount of stress introduced in the tissue. The unatural elongation was expected, as the linear stress-strain model is not valid for large forces.

The LR Spring model from section 5.7 was constructed specifically for large geometries such as the cardiac geometry. The deformation equilibrium was divided into two areas of realistic and less realistic deformation. The assumption that the most important part of the deformation is always close to the interaction point was proven to be valid for surgeons. The deformation was evaluated as very realistic and believable.

The main problem with the Spring Mass based models is that part of the geometry can relatively easy flip into itself (see section 5.10) when the geometry is compressed. The surgeons experienced this problem, but did not recognize this as a serious problem because the only clear visible sympton is un-even shading. The problem is serious though, becase the topology, and therebye behavior, of the tissue changes.

## 10.2 Dynamic or static model

As observed in section 2.6 the vibrations of the materials are not noticeable because of the characteristics of the interaction with the tissue. The dynamic and static behavior of a Spring Mass model was evaluated with the surgeons. The static system was evaluated as the most realistic - vibrations are simply not noticeable when working on objects of such small scale with a controlled interaction.

## 10.3 Deformation animation

The deformation animation denotes the behavior over time. Deformation animation includes convergence through iteration and time dependant movement. In the case of static deformation, animation is naturally part of the movement of interaction points. The iterative algorithms used to calculate the deformations are also very visible though, because equilibrium cannot be established within one frame if the interacting forces are of a certain size.

Through evaluation with the surgeons it was found to be important to get a fast convergence towards equilibrium to achieve a realistic behavior. The basic Spring Mass model and FEM gave a slow convergence for large models such as the heart geometry. If the Spring Mass model was combined with an iterative relaxation, the convergence was faster and gave the impression of a much more realistic tissue.

With the LR Spring model the deformation animation was very fast in the area of interest because the pure Spring Mass dynamics combined with relaxation in a small area gives a very fast convergence. In the rest of the organ a relatively fast convergence was realized through relaxation initially driven by movement of nodes in the area of interest. Again the LR Spring model was evaluated as more realistic in comparison to the other models.

The LR Spring Mass (with a cut-off at depth four and one relaxation iteration pr frame) gave a frame rate of 20 fps. The surgeons found it to be fast enough for interactive use.

## 10.4 Interaction

The abstract tools from chapter 7 were generally useful for the surgeons, as they were more suited for pre-operative planning than completely realistic simulation of surgical instruments. The grab-tool was successful because there was no collision detection that could introduce unwanted forces into the tissue. It was validated that the surgeons did not need pushing and probing instruments, but only pulling and cutting. In the current implementation there is only one type of grab-tool. The surgeons would like other grab-tools with different areas of interaction - simulating instruments of different sizes and shapes.

In the current implementation there is only one point of interaction. The surgeon actually needs many more points of interaction to do a complete surgical procedure. In reality two surgeons do the procedure in cooperation, holding and manipulating instruments.

The cutting did not introduce any collision response until it was activated. This was helpful for the interaction with the tissue via a mouse. In the current implementation the cut-sweep was a triangle of a constant size, it was noted that the triangle should be re-sizable to make smaller or more precise cuts. The cutting in itself was evaluated as realistic even though there was no tissue response. Tissue response would add to the realism, but was not important as part of the pre-operative tool.

As part of the construction of the Connected Surface structure in section 5.9.4, I assumed that cuts were perpendicular to the surface, and often go through the whole tissue. This was evaluated as a valid assumption.

## 10.5 Simplifications

Gravity was neglected from the LR Spring model to simulate point interactions faster. The missing simulation of gravity was not an issue for the surgeons, as gravity is not an issue in real surgery. Of course gravity is evident in a surgical procedure, but the surgeon has learned how to abstract away from it. When compared to the deformations due to pushing, pulling and cutting, gravity seems to be a negligible force in the heart of a child.

## 10.6 Categories of Use

It was discussed how a surgical simulator could be used in relation to the categories of use presented in section 2.8. The surgeons could generally easily see the use of the surgical simulator as a pre-operative tool. The surgeons were very positive towards the possibility of using the surgical simulator as a tool for analysis of the surgical procedure. It was pointed out by the surgeons that the simulation would give them a more natural presentation of the heart than a pure geometrical presentation would. Instead of walking into the model to look at something, the surgeon can cut the model open and observe the model, just like it would look in a actual surgical situation.

The surgeons could easily imagine the surgical simulator as part of a surgical training, though this was not explicitely part of this thesis.

With respect to skill assesment, Kromann said that this would not be realistic in Denmark because the choice of apprentice is a very personal one.

# Chapter 11

# Conclusion

The goal of this thesis was to investigate the possibility of a realtime surgical simulator specifically for congenital cardiac diseases. Previous surgical simulators based on Spring Mass models or FEM have simulated organs with relative simple morphology. This thesis represents the first steps in the direction of a realtime surgical simulator that can simulate geometrically complex organs, such as the heart. Specifically for this use, the LR Spring Mass model was developed.

A number of Spring Mass inspired models and a Conjugate Gradient FEM were compared for rate of convergence and equilibrium over a series of interactions through time. It was experienced that both Spring Mass and FEM had a convergence that was too slow for a credible realtime simulation for geometrically large models. With an iterative relaxation algorithm used on a Spring Mass based deformation the convergence of force in large geometries is more believable. As part of the thesis a number of different techniques for surgical simulation have been implemented and evaluated with surgeons. The LR Spring Mass model was evaluated favorable in comparison to previous models.

Apart from technical aspects as presented in this thesis I have learned a lot from the cooperation with the surgeons. I have had the chance to learn about a completely different field. It has been a highly motivating factor that I cooperated with real people with a need for a tool to analyze real situations. A surgical simulator has the potential to help the surgeon make vital decisions.

# Chapter 12

# Future Research

The field of surgical simulation is large and many different techniques have been suggested. In this thesis I have chosen to look at some of the most popular techniques of surgical simulation, but a large amount of alternatives and small variations exist.

## 12.1 Further evaluation with surgeons

Generally a future research would continue the evaluation with the surgeons, and extend to evaluation of complete surgical procedures in the simulation.

A specific next step in the surgical simulator would be to simulate patches. In section2.5 about surgical procedures we saw that patches are often used to reconstruct some part of the heart morphology. As patches are in themselves soft materials, we can simulate them with some of the same techniques as soft tissue. Working with patches includes cutting them into some shape and stitching them onto the tissue.

If the surgical simulator is to be used through a whole surgical procedure we will also need to simulate suturing in some way. One idea proposed by the surgeons was to simplify this operations within the perspective of abstract tools. For an experienced surgeon suturing is not difficult, and in the simulator he could simply define connections between positions on the tissue.

## 12.2 Validation

According to Gibson [27] three kinds of validation exist: Expert evaluation, comparison to more precise elastic models and formal experimental validation. In this thesis I have validated the surgical simulator with expert evaluation and comparison to FEM.

To validate the surgical simulator for a specific case of use we need one further step of validation. We need to formally validate that the surgical simulator is actually useful for the specific case of use chosen. This kind of validation

is expected to be the next big step in the field of surgical simulation [71, 43]. Richard Satava has been the primary spokes person for the validation of surgical simulators in training scenarios. A surgical simulator can be tested for validity and reliability with a series of tests, see appendix C.

## 12.3 Experimental comparison of models

The experimental comparison of elastic models introduced in this thesis has been tested on the model introduced in this thesis. A range of variations and very different alternatives exists for the calculation of deformation, but often they are not compared in a rigorous manner. A future research would include a comparison of a larger range of models so they could be ordered in hierarchies based on their different attributes.

A larger comparison would be based on a range of test interactions derived from real surgical procedures. In relation to a large number of test interactions we might specifically for the springs mass model divide the springs into categories as in the case of the Connected Surfaces structure and optimize for spring stiffness values for each category of springs. We have some indication from the parameter optimization of the Connected Surfaces that a homogeneous set up of springs is not the same as a homogeneous set up of the FEM, simply because springs are locally defined.

## 12.4 Seeded iterative models

Both the CG FEM and the Quasi Static algorithm are iterative methods that can be seeded with a solution guess. In the thesis we have only discussed a simple seeding with the nodal positions of the previous frame, but in some situations this strategy is not optimal.

One extreme case when this simple strategy is not optimal is when an interaction point is released. We would expect the tissue to return to the initial configuration of points, and it will, but characteristics in especially the Spring Mass model results in a very slow convergence. In future work we could take advantage of the fact that we can seed the iterative algorithms, e.g. with the initial configuration after the release of interactions points. If multiple interactions points can be active at the same time (using multiple instruments) and only one interaction point was released, we could set the positions of the nodes based on their distance to the released interaction point. I.e. nodes close to the released interaction point would be moved to their initial configuration while nodes further away would be seeded with their current location.

A more general definition of the seeded iterative models would be to base the deformation on two elastic models; a precise iterative model and a less precise heuristic model. The less precise heuristic model would quickly find a near-optimal nodal configuration, and the precise iterative method would refine the nodal configuration.

Such a strategy for deformation has not been investigated previously beyond one method; the 3D chainmail method combines a heuristic elastic model l with a relaxation algorithm [25]. The chainmail algorithm is not physically based though.

## 12.5 Specialized physical models

The basic Spring Mass model and FEM are very general in nature, and can be used to simulate a range of different material behavior. The LR Spring Mass model with a Connected Surfaces structure created as part of this thesis used domain specific knowledge to create a better elastic model for the specific case of surgical simulation in a child's heart. Future research within surgical simulation should use the domain specific knowledge to a greater extent, both with respect to material behavior of tissue and typical interaction in a surgical procedure. The acknowledgment of domain specific knowledge will allow us to abstract unimportant parts of reality away and focus on those parts of reality that are vital to simulation of surgical procedures.

# Bibliography


[1] Ackerman, M. J. *Accessing the Visible Human Project*. D-Lib Magazine, October 1995.

[2] Agerwall, R., et. al. *Special Visual Effects for Surgical Simulation: Cauterization, Irrigation and Suction*. Medicine Meets Virtual Reality 11, 2003, pp. 1-3.

[3] Bielser, D. and Gross, M. H. *Interactive Simulation of Surgical Cuts*. Proceedings of Pacific Graphics 2000, IEEE Computer Society Press, 2000, pp. 116-125.

[4] Bielser, D., et al. *Interactive Cuts through 3-Dimensional Soft Tissue*. Computer Graphics Forum (Eurographics '99 Proc.) 18, no. 3, 1999, pp. 31-38.

[5] Bishop, B., et. al. *SPARTA: Simulation of Physics on a Real-Time Architecture*. Proc. of 10th Great Lakes Symposium on VLSI, Evenston, IL, March 2000.

[6] Bower, A. F. *Online notes for course: Advanced Mechanics of Solids*, www.engin.brown.edu/courses/EN175/notes.htm (30/05-2003).

[7] Brown, J., et. al. *Real-Time simulation of Deformable Objects: Tools and Application*. In Proceedings of Computer Animation 2001, 2001, pp. 228-236.

[8] Bruyns, C., Montgomery, K. and Wildermuth, S. *A Virtual Environment for Simulated Rat Dissection*. IEEE Visualization 2001, 2001.

[9] Bryuns, C. D. and Montgomery, K. *Generalized Interactions Using Virtual Tools within the Spring Framework: Cutting*. Medicine Meets Virtual Reality (MMVR02), 2002,

[10] Bryuns, C. D. and Montgomery, K. *Generalized Interactions Using Virtual Tools within the Spring Framework: Probing, Piercing, Cauterizing and Ablating*. Medicine Meets Virtual Reality (MMVR02), 2002

[11] Cakmak, H. K., Kühnapfel U. *Animation and Simulation Techniques for VR-Training Systems in Endoscopic Surgery*. Eurographics Workshop on Animation and Simulation '2000 (EGCAS '2000), 2000, pp. 173-185.

[12] Casson, F. B. de and Laugier, C.: *Modelling the dynamics of a human liver for a minimally invasive surgery simulator.* Medical Image Computing and Computer Assisted Intervention 1999 (MICCAI'99), vol. 1679 of Lecture Notes in Computer Science, 1999, pp. 1156-1165.

[13] Cincinatti Childrens Hospital Medical Center. *The Heart Center Encyclopedia* www.cincinnatichildrens.org/health/heart-encyplodeia/default.htm (30/05-2003).

[14] Cotin, Stephane et. al. *A Hybrid Elastic Model allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation.* Visual Copmuter journal, Vol 16, No 8, 2000, pp. 437-452.

[15] Cotin, Stephane et al. *Efficient Linear Elastic Models of Soft Tissue for real-time surgery simulation.* Rapport de recherche n3510, October 1998.

[16] Cotin, Stephane et al. *Real-time elastic deformations of soft tissues for surgery simulation.* IEEE Transaction on Visualization and Computer Graphics, 5(1), 1999, pp. 62-73.

[17] Cover, S. A., et. al. *Interactively Deformable Models for Surgery Simulation.* Proceedings of ICRA, 1993, pp. 68-75.

[18] Delingette, H., et. al. *A Craniofacial Surgery Simulation Testbed.* Visualization in Biomedical Computing (VBC'94), 1994, pp. 607-618.

[19] Delingette, H. *Towards Realistic Soft Tissue Modeling in Medical Simulation.* Proceedings of the IEEE: Speciale Issue on Surgery Simulation, 1998, pp. 512-523.

[20] Deussen, O., et. al. *Using Simulated Annealing to Obtain Good Nodal Approximations of Deformable Bodies.* Computer Animation and Simulation '95, 1995, pp. 30-43.

[21] Freitag, L. A. and Ollivier-Gooch, C. *A Cost/Benifit Analysis of Simplicial Mesh Improvement Techniques as Measured by Solution Efficientcy.* International Journal of Computational Geometry and Applications, vol 10, no 4, 2000, pp. 361-382.

[22] Ganovelli, F. and O'Sullivan, C. *Animating cuts with on-the-fly-re-meshing.* Eurographics 2001, 2001, pp. 243-247.

[23] Gibson, S. F. F. *3D ChainMail: a Fast Algorithm for Deforming Volumetric Objects.* Proceedings of the Symposium on Interactive 3D Graphics, ACM Press, 1997, pp. 149-154.

[24] Gibson, S. F. F and Mirtich, B. *A Survey of Deformable Modeling in Computer Graphics.* MERL Technical Report TR-97-19, November 1997.

[25] Gibson, S. F. F. *Beyond Volume Rendering: Visualization, Haptic Exploration, and Physical Modeling of Voxelbased Objects.* MERL Technical Report 95-04, 1995.

[26] Gibson, S. *Simulating Arthroscopic Knee Surgery using Volumetric Object Representations, Real-Time Volume Rendering and Haptic Feedback.* In first Joint Conference on Computer Vision, Virtual Reality, and Robotics in Medicine and Medical Robotics and Computer Assisted Surgery, 1997, pp. 369-378.

[27] Gibson, S. F. F. *Volume Deformation: Modeling shape changes in Sampled Volumes.* SIGGRAPH 1999.

[28] Gibson, S. F. F. *Using Linked Volumes to Model Object Collisions, Deformation, Cutting, Carving, and Joining.* IEEE Transactions on Visualization and Computer Graphics, October-December 1999,Vol. 5, No. 4, 1999, pp. 333-348.

[29] Gorman, P. J., et. al. *Simulation and Virtual Reality in Surgical Education.* Archives of Surgery, 1999, pp. 1203-1208.

[30] Helbing, D., et. al. *Simulating dynamical features of escape panic.* Nature 407, 2000, pp. 487-490.

[31] Hang, S. *Tetgen, The Quality Tetrahedral Mesh Generator,* http://tetgen.berlios.de/ (08/05-2003)

[32] Hansen, K. V. and Larsen, O. V. *Using Region-of-interest based Finite Element Modelling for Brain-Surgery Simulation.* Procedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention, 1998, pp. 305-316.

[33] Harder, M. et. al. *Comparing a Simplified FEM Approach with the Mass-Spring Model for Surgery Simulation.* Medicine Meets Virtual Reality 11, 2003, pp. 103-109.

[34] Harris, J. and James, G. *Physically-Based Simulation on Graphics Hardware.* GameDevelopers Conference, 2003.

[35] Holbrey, P. R. and Bulpitt, A. J. *Metrics and Motion Analysis for Assessment of Surgical Skills.* Medicine Meets Virtual Reality 11, 2003, pp. 124-126.

[36] Jacobsen, T. *Advanced Character Physics* Proceedings, Game Developer's Conference 2001, San Jose 2001.

[37] Kaufmann, C., et. al. *First Steps in Eliminating the Need for Animals and Cadavers in Advanced Trauma Life Support.* Medical Image Computing and Computer-Assisted Intervention (MICCAI), 2000, pp. 618-623-

[38] Keeve, E., Girod, S. and Girod, B. *Craniofacial Surgery Simulation*. Proceedings of the 4th International Conference on Visualization in Biomedical Copmuting (VBC'96). 1996, pp. 541-546.

[39] Kincaid, D. and Cheney, W. *Numerical Analysis: Mathematics of Scientific Computing*, Third Edition.

[40] Kühnapfel, U., Cakmak, H. K and Maass, H. *Endoscopic surgery training using virtual reality and deformable tissue simulation.* Computers & Graphics, Volume 24, Issue 5, 2000, pp. 617-682.

[41] Larman, C. *Applying UML and Patterns. An Introduction to Object-Oriented Analysis and Design*. 1998, Prentice Hall.

[42] Lin, W. and Robb, A. *Dynamic Volume Texture Mapping and Model Deformation for Visually Realistic Surgical Simulation*. Medicine Meets Virtual Reality 7, 1999, pp. 198-204.

[43] Magee, J. H. *Validation of Medical Modeling & Simulation Training Devices and Systems*. Medicine Meets Virtual Reality 11, 2003, pp. 196-198.

[44] Mathiassen, L., et. al. *Object Orienteret Analyse og Design*, 2. udgave, 1998, Forlaget Marko.

[45] McCloy, R. and Stone, R. *Science, medicine, and the future. Virtual reality in surgery.* BMJ2001;323, 2002, pp. 912-915.

[46] Michalewicz, Z. and Fogel, D. B. *How to Solve It: Modern Heuristics*. Corrected Second Printing 2000, Springer-Verlag.

[47] Mitchell, Melanie: *An Introduction to Genetic Algorithms*. Sixth printing 1999, A Bradford Book, MIT Press.

[48] Montgomery, K. *Enabling Technologies in Surgical Simulation: When will the future be here*. Presentation at TATRC's third annual Principal Investigators Review. 2003.

[49] Montgomery, K. et al. *Surgical Simulator for Hysteroscopy: A Case Study of Visualization in Surgerical Training*. 12th IEEE Visualization 2001 (VIS2001), 2001.

[50] Montgomery, K. and Bruyns, C. *Virtual Instruments: A Generalized Implementation*. Medicine Meets Virtual Reality 11, 2003, pp. 210-215.

[51] Mor, A. B: *Progressive Cutting with Minimal New Element Creation of Soft Tissue Models for Interactive Surgical SImulation*. phd thesis. The Robotics Institute Carnegie Melloon University. October 11, 2001.

[52] Nash, M. P. and Hunter, P.J. *Heart Mechanics Using Mathematical Modelling.* Proceedings of the 2nd Postgraduate Conference for Engineering and Technology Students, 1995.

[53] Nebel, J. *Soft tissue Modelling from 3D scanned data.* Proceedings of Deform2000, also in Deformable Avatars, N. Magnenat-Thalmann and D. Thalmann (editors), Kluwer, 2001, pp. 85-97.

[54] Nedel, P. L. and Thalmann, D. *Real Time Muscle Deformations Using Mass-Spring Systems.* Proceedings of Computer Graphics International'98 (CGI'98), IEEE Computer Society Press. 1998, pp. 156-165.

[55] Neider, J., et al. *OpenGL Programming Guide*, 1. release, 1994, Addison-Wesley Publishing Company.

[56] Nielsen, M. B. *Fast Finite Elements for Surgery Simulation.* Medicine Meets Virtual Reality 5 (MMVR5'97), 1997. pp. 395-400.

[57] Nielsen, M. B., et al. *VR Simulation of Abdominal Trauma Surgery.* Medicine Meets Virtual Reality 6, 1998, pp. 117-123.

[58] Nielsen, M. B. *Finite Element Modeling in Surgery Simulation.* Proceedings of the IEEE: Special Issue on Virtual & Augmented Reality in Medicine, 86(3), March 1998, pp. 524-530.

[59] Nienhuys, H. and Stappen, A. F. van der. *A surgery simulation supporting cuts and finite element deformation.* Medical Image Computing and Computer-Assisted Intervention, 2001, pp. 153-160.

[60] Nienhuys, H. and Stappen. A. F. van der. *Combining finite element deformation with cutting for surgery simulations.* EuroGraphics Short Presentations, 2000, pp. 43-52.

[61] Nienhuys, H. and Stappen, A. F. van der. *Supporting cuts and finite element deformation in interactive surgery simulation.* Tech-report, Utrecht University, Institute for Information and Computing Sciences, June, 2001.

[62] Nienhuys, Han-Wen and Stappen, A. F. van der. *A simple mesh data structure with application in surgery.* EuroImage International Conference on Augmented, Virtual Environments and Three-Dimensional Imaging, 2001, pp. 148-151.

[63] Nienhuys, H. and Stappen, A. F. van der. *A Delaunay approach to interactive cutting in triangulated surfaces.* Fifth International Workshop Foundations of Robotics (WAFR 2002), 2002.

[64] Holden, A. *The beating heart of virtual engineering.* Scientific Computing World Oct/Nov 2000, pp. 26-28.

[65] Park, J. *Shape Retaining Chain Linked Model for Real-time Volume Haptic Rendering.* Volume Visualization and Graphics 2002, 2002, pp. 65-72.

[66] Provot, X. *Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior.* Graphics Interface '95, 1995. pp. 147-154.

[67] Reddy, J.N. *Theory of Elasticity, Supplementary Notes.* Department of Mechanical Engineering. http://ceprofs.tamu.edu/jreddy/MEMA601/LectureNotes.pdf (30/05-2003).

[68] Salb, T., et. al. *Haptic Based Risk Potential Mediation for Surgery Simulation.* Proceedings of 1. International Workshop on Haptics Devices in Medical Applications (HDMA), within the scope of CARS 1999.

[69] Salb, T., et. al. *Preoperative planning and training simulation for risk reducing surgery.* Proceedings of International Training and Education Conference (ITEC) 1999.

[70] Satava, R. *Medical Virtual Reality:* The *current status of the future.* Proceedings Medicine Meets Virtual Reality (MMVR IV). 1996.

[71] Satava, R. M. *Report on the Metrics for Objective Assessment of Surgical Skills Worshop.* Presentation at TATRC's third annual Principal Investigators Review at MMVR11, 2003.

[72] Shewchuk, J. R. *An Introduction to The Conjugate Gradient Method without the Agonizing Pain*", Edition 1 1/4.

[73] Terzopoulos, D. *Analysis and Synthesis of Facial Image Sequences Using Physical and Anatomical Models.* IEEE Transaction on Pattern Analysis and Machine Intelligence vol 15, No 6, June 1993.

[74] Sørensen, T. S., et. al. *Morphological Visualization in Congenital Heart Disease: Virtual, Three-Dimensional Reconstruction from Magnetic Resonance Imaging.* Cardiology in the Young 2003, In Press, 2003.

[75] Terzopoulos, D., et. al. *Elastically deformable models.* Computer Graphics, 21(4), Proceedings in ACM SIGGRAPH'87 Conference, 1987, pp. 205-214.

[76] Therkildsen, S. V. and Sørensen, T. S. *Visualization of the cardiovascular System in a Virtual Reality Environment for Preoperative Planning of Cardiac Surgery.* Masters thesis. Computer Science Department, Aarhus University, Januar 2000.

[77] Wilhelms, J. and Gelder, A. Van. *Anatomically Based Modeling.* Proceedings of SIGGRAPH 97, 1997, pp. 173-180.

[78] Wu, Yi et. al. *Deformable Surfaces using Physically-based Particle Systems.* Proceedings of VHCGI95, 1995.

[79] Young D. Hugh, et al. *Sears and Zemansky's University Physics*, tenth edition, 2000.

# Appendix A

# FEM Details

## A.1 $K^e$ is symmetric

Pr definition $C$ is symmetric. Therefor $K^e$ is also symmetric:

$$\begin{aligned}(K^e)^T &= (B^{eT} C\, B^e)^T V^e \\ &= (C\, B^e)^T B^e V^e \\ &= B^{eT} C^T B^e V^e \\ &= B^{eT} C\, B^e V^e \\ &= K^e\end{aligned}$$

## A.2 $B^e$ matrix

The $B^e$ matrix becomes:

$$B^e = \begin{bmatrix} b_1 & 0 & 0 & b_2 & 0 & 0 & b_3 & 0 & 0 & b_4 & 0 & 0 \\ 0 & c_1 & 0 & 0 & c_2 & 0 & 0 & c_3 & 0 & 0 & c_4 & 0 \\ 0 & 0 & d_1 & 0 & 0 & d_2 & 0 & 0 & d_3 & 0 & 0 & d_4 \\ c_1 & b_1 & 0 & c_2 & b_2 & 0 & c_3 & b_3 & 0 & c_4 & d_4 & 0 \\ 0 & d_1 & c_1 & 0 & d_2 & c_2 & 0 & d_3 & c_3 & 0 & d_4 & c_4 \\ d_1 & 0 & b_1 & d_2 & 0 & b_2 & d_3 & 0 & b_3 & d_4 & 0 & b_4 \end{bmatrix}$$

# Appendix B

# Surgical Simulator Implementation

## B.1 CD-ROM

The source code is available on the CD-ROM coming with this thesis. Binaries and movies are also available on the CD-ROM.

## B.2 Compilation

The surgical Simulator compiles under gcc 3.2 for cygwin. The implementation depends on OpenGL, GLUT and MTL.

To compile the entire surgical simulator type:

```
make
```

The surgical simulator is as standard compiled to a file named:

```
sim.exe
```

Without any parameters the simulator will run a default surgical simulator setup. Text files (Init files) are given as argument to the executable to set up a certain geometry and elastic model.

If two init files are given, the Elastic Models are compared to each other. The reference model controls the interaction of the target model. With view number 6 the difference on a nodal basis is visualized. E.g.:

```
sim.exe InitTestFEM InitTest2
```

If two init files and the command EA is given, the EA algorithm is run to find optimal parameters for the second elastic object. E.g.:

```
sim.exe InitTestFEM InitTest2 EA
```

## B.3 Init files

A range of initialization files are avalible, init files are named Init*. If the init files are for the wall test example they are named InitTest*. The heart is named InitHeart.

## B.4 Interaction

The camera is moved in space with the keyboard:

**w** forward

**s** backward

**a** left

**d** right

The mouse controls the position of the abstract tool in relation to the current camera position. If the left mouse button is pressed the mouse controls the distance of the tool to the camera. If left shift and left mouse button is pressed, the orientation of the view and tool is changed with mouse movements.

The Tools are selected with:

**z** GrabTool

**x** ProbeTool (experimental)

**c** CutTool

## B.5 View

Select keyboard the following views:

**1** Gouraud shading

**2** Flat shading

**3** Wireframe

**4** Wireframe + gouraud shading

**5** connectedEdges (for ConnectedEdges structure)

**6** Comparison of two ElasticObjects

## B.6 Implementation details

If the reader should choose to look at the source code, the following is a guide for the sources files. The implementation is written in C++ and can be compiled with gcc 3.2 under cygwin. The implementation depends on OpenGL and GLUT for visualization and MTL for basic matrix operations.

SurgerySim.cpp The main file. Contains code for OpenGL initialization, setup of camera, drawing of some of the scene. Keyboard and mouse interaction is reacted to and sent to the ElasticObjects. It is also the responsibility of the SurgerySim.cpp to react to arguments to the executable, load the init files containing initializing information and instantiate the correct ElasticObjects and Geometry objects. If an EA is wanted the EA class is instantiated.

ElasticObject.h/cpp Defines the basic Tetrahedron class, Geometry class, Ray class (for connection of ConnectedSurfaces), Node class, Hold class, Triangle class and ElasticObject class. The ElasticObject defines virtual function with basic drawing capabilities.

SpringMassObject.h/cpp Defines the ParticleNode class, Edge class (is actually a Spring, as this is the only Edge class there is because the FEM model does not use edge classes), EdgeTriangle class (a special class for triangles that can have edges - used with the Spring Mass models) and SpringMass class.

QuasiStaticObject.h/cpp Defines the QuasiStaticNode class and the QuasiStaticObject class.

FEMObject.h/cpp Defines the DisplacementNode class, the FEMTetrahedron class and FEMObject class

matrixtypes.h Defines matrix types used.

matrixutil.h A small collection of functions for matrix manipulation

Iterative.h Defines IterativeSolver class, StepestDecent class and ConjugateGradient class.

Surface.h/cpp Defines the Surface class.

ConnectedSurface.h/cpp Defines the ConnectedSurfaces class.

octree.h/cpp Defines the Octree class used for acceleration in ConnectedSurface class.

TetrahedronMesh.h/cpp Defines the TetrahedronMesh class.

Tool.h/cpp The Tool hierarchi: Tool class, GrabTool class, CutTool class and ProbeTool class.

ElasticComparison.h/cpp Defines the ElasticComparison class.

EA.h Defines the EA class, the Chromosome class and it's specializations. Is edited for different runs of EA.

l3ds.h/cpp Used for loading 3DS files, slight alterations of the original source code (the OBJ and SMESH file formats are easily loadable and are loaded in the Surface class and TetrahedronMesh respectively)

preformer.cpp A collection of classes for 3d point definition and manipulation. mimics some of the Performer library functions and classes for easy porting.

timeexp.h/cpp Functions for time related evaluation.

# Appendix C

# Validation and Reliability

It has been recognized [43] that the next big step in surgical simulation is a formal verification of the usefulness of training with surgery simulation. In [71], Richard Satava presented the progress in the Metrics for Objective Assesment of Surgical Skills Worshop. A Surgical Simulation system must be able to show Validity and Reliability. The kind of validity and reliability required is specifically for the training scenario and use of surgical simulation.

The bio-mechanical behaviors of the model are not evaluated directly, but the outcome of a training situation with the surgical simulation is.

The Surgical Simulator built as part of this thesis was not tested on real students. And the discussion of validity and reliability will be a hypothetical one with arguments from the surgeons.

## C.1 Validity

Validity is defined as :

**Face** Experts review the tests to see if they seem appropriate "on their face value"

**Content** Experts perform a detailed examination of the contents of the tests to determine if they are appropriate and situation specific.

**Construct** The determination of the degree to which the test captures the hypothetical quality it was designed to measure.

**Concurrent** The realtionship of the new test score (and those) whose performance has been evaluated in actual working conditions.

**Predictive** Determining the extent to which the scores on a test are predictive of actual performance.

It is important to note that this kind of validity does not explicitly validate the bio-mechanical model of the simulator. It validates that students learn something useful from the surgical simulator.

## C.2 Reliability

Reliability of a simulator is defined as:

**Inter-rate** Determining the extent to which two different evaluators (raters) score the same test.

**Test-retest** Reliability of a test by administering it two (or more) times to the same persons and obtainin a correlation between the scores on each testing

## C.3 Taxonomy

A Taxonomy of what to train is also important. [71] has a hierarchical presentation of goals in training: Abilities, Skills, Tasks and Procedures.

**Ability** The state or condition of being capable; aptitude; competence; capability; power to do something, physical, mental, legal etc.

**Skill** A developed proficiency or dexterity in some art, craft, or the like; deftness in execution or performance; a trade or craft requirering special traning for competence or expertness in its practice.

**Task** A piece of work imposed upon a person by another; a piece of work to be done; that which duty or necessity imposes; an undertaking; a burdensome, difficult or unpleasant chore or duty; a difficult or tedious undertaking.

**Procedure** A series of steps taken to accomplish an end; a manner of proceeding; a way of performing or effecting something.

An ability is the simplest possible training; psycho-motor, visio-spatial, perception or haptic abilities. A Skill is a classic surgery skill such as e.g. instrument handling, bimanual dexterity, Knot tying, Tissue handling and cutting and several abilities. A Task is a classic surgery task such tissue extraction, closure etc. A Procedure is a surgery procedure such as a VCD or an ACD presented in section 2.5.

When building a simulator one must be very carefull about what level of such taxonomy the training focuses on.

# Appendix D

# Evaluation from Surgeons