Haptic Interaction with Deformable Objects

Igor Peterlík

PhD Thesis Proposal
The Faculty of Informatics, Masaryk University in Brno

Brno, 2006

I certify that this text is adequate as a thesis proposal. I agree with this proposal.

doc. RNDr. Luděk Matyska, CSc., advisor
Brno, December 2006

Acknowledgments

I would like to express my deep thanks to doc. Luděk Matyska and dr. Eva Hladká for supporting me in my work and motivating me. I would also like to thank to all fellows from the Laboratory of Advances Networking Technologies at the Faculty of Informatics, Masaryk University, for all given advice.

Igor Peterlík

Contents

Chapter 1

Introduction

Modelling of physical systems via computer simulation has been of great interest since the era of the first computers. There are many phenomena, which cannot be directly observed in real life (e. g. the molecular docking, black hole behaviour) or their measurement is too dangerous and expensive to be experimentally undertaken (e. g. nuclear explosions, brain operations).

The overwhelming progress in computer science enabled still larger and more complex systems to be analysed. Moreover, the visualisation of results has become very important part of the modelling thanks to computer graphics. A much more advanced phase of the research has been introduced by employing the virtual reality resulting in real-time simulations, when the user becomes really involved in the phenomenon being studied.

One of the interesting areas of research is represented by the soft tissue modelling with application in surgical simulators. The motivation of the research is twofold: first, the medical training using virtual environment would enable the surgeons and other staff to learn and practise virtual operations with no risk to patient's health. Second, the operation planning would improve the treatment of non-standard situations and procedures, as the operation could be first performed on a virtual model of a patient with a special or complicated disease. Creating an accurate model of the patient's body or its relevant part according to some scanning technologies such as nuclear magnetic resonance can be very helpful, since many unexpected issues can be addressed in the preparation phase and the operation can be planned with respect to the real state of the patient.

Besides the high-quality visualisation, the main and inevitable component of the virtual simulation of soft tissue is the haptic interaction which allows the user to "touch and feel" the virtual objects. This imposes much higher requirements on the speed and accuracy of computations when simulating the behaviour of the modelled objects. e. g. whereas the refresh rate needed for realistic visualisation is about 25 Hz, the refresh rate required in haptic rendering is usually above 1 kHz due to the much higher tactile sensitivity of the human perception.

On the other hand, the models should be convincing and therefore based on real physics. The relations from the theory of elasticity are usually employed when establishing the mathematical formulation of the problem that is finally solved by some complex method such as finite elements. It is a well-known fact that performing this computations within haptic loop is far beyond the capabilities of nowadays computers.

There have been several attempts to combine the high-fidelity of the model with the requirements of the haptic interaction. The main approaches are usually represented by some simplification of the model (e. g. linearisation) or restriction of the interaction (e. g. the laparoscopic surgery). Our goal is to propose a novel approach, which would allow haptic interaction with complex models ensuring the realistic behaviour of the simulation. The approach is based on parallel precomputation of the state space of the interaction. We would also like to address the issues which appears when considering the action such as cutting or tearing of the tissue which are however crucial for real usability of the surgical simulator.

This proposal is organised as follows. In chapter 2, we introduce the basic terms from the area of deformation modelling, especially focused on the finite element method. In chapter 3, we present the current approaches to the soft tissue modelling as well as the solution of the large systems of algebraic equations, which arise in solution of the deformation problems. In the chapter 4, we present the list our objectives and then in the chapter 5, we shortly describe the applied approaches. Finally, in the chapter 6, we propose the time schedule of our work.

Chapter 2

Preliminaries

First, we introduce the basic terms and methods used in the area of the deformation modelling. For the sake of the completeness, we present a brief overview of non-physically and physically based approaches. Then we focus on finite element method, which is the core of our work, and present the description of its application in soft-tissue modelling.

2.1 Deformation Modelling Approaches

The main goal of the deformation analyses and modelling is the computation of a new geometry of a deformable body after external or internal forces have been applied. The main area of application is in design and construction of buildings and machines, crash test simulations, soft tissue modelling. There are two basic approaches — non–physical and physical modelling.

2.1.1 Non-physical modelling

Non-physical methods for modelling of deformable objects are usually based on pure heuristic geometric techniques or use a sort of simplified physical principles to achieve the reality like effect [1]. Two widespread methods are:

- Spline modelling where both planar and 3D curves and surfaces are represented by a set of control points. The shape of complex objects is then modified by varying the position of the control points. [2]
- Free form deformations where the the main idea is to deform the shape of an object by deforming the space in which it is embedded. [3, 4]

2.1.2 Physical Modelling

Physical modelling is based on solving problems from the *theory of elasticity* under consideration of material properties. The realistic simulation of physical bodies and their behaviour provided by physical modelling is based on solving governing partial differential equations by suitable numerical technique. In following list we shortly describe several methods for physically based numerical modelling.

Mass-spring method The mass is concentrated in a number of nodes which are connected by springs. When a force is applied to a node, it starts to move and pass the force via springs to the neighbouring nodes. [5, 6]

The computations of deformation in this model are much less expensive comparing to the other models and allow haptic real-time interaction. On the other hand the main disadvantage of mass-spring models is their insufficient approximation of true material properties, since they are not directly based on the equations of continuum mechanics.

Finite difference method The continuous derivative is replaced with a finite difference approximation in considered domain. It is usually applied on cubic grids, where the geometry of the problem is regular. Since the discretisation of objects with irregular geometry becomes extremely dense, the method is not suitable for general modelling of objects.

Boundary element method (BEM) In this approach, the differential problem is converted to the integral form, where under special conditions, the integration over the volumetric domain can be substituted by the integration over its boundary Γ. Deformation model using BEM is described in [7].

Since the reduction of integration requires homogeneity of material as well as the BEM does not allow "volumetric" operations such as cutting and suturing, it is again not suitable for construction of surgical simulators.

Finite element method Finite elements are widely used for modelling of soft tissues, since they are directly based on the theory of the elasticity and provide very good approximation for the complex geometries. The method generally consists of discretisation of the domain mathematical formulation of the governing equation over the elements and results in a large system of algebraic equations [8].

The other less known approaches are methods using precomputed Green Functions [9, 10], which can be applied on a special kind of problems, *finite sphere models* [11] and *Long Element Method* described in [12] that is FE based method using special domain discretisation.

The physically-based methods are computationally expensive, however, they provide better properties and fidelity of the simulation comparing to the non-physical techniques. Further, the finite element method seems to be superior to the other methods when modelling complex bodies with non-trivial physical properties. It is also suitable when considering the volumetric operations as cutting and tearing, since it models the entire body of the deformable object. Therefore we consider this method as the most suitable for modelling the soft tissues deformations, where the accurate simulation of complicated geometries is of a great interest.

2.2 Elasticity Theory

2.2.1 Physical Representation of Deformations

In our work, we focus on the physical modelling, which is based on the theory of elasticity. Therefore in this section, we present several fundamental relations and terms used in the deformation modelling.

Having a deformable body, its full description consists of:

- the geometry — the shape of the body, e. g. the coordinates of particles of the body in Cartesian coordinate system

- the physical properties i. e. the type or the material, gradients of density of some other parameters
- the boundary conditions — the points or areas where the body is fixed in space somehow

There are two basic physical quantities taking part in the process of deformation — *forces* and *displacement*.

First, having the deformation that moves a particle at position $\mathbf{x}$ to a new position $\bar{\mathbf{x}}$, the displacement $\mathbf{u}(\mathbf{x})$ is defined as the difference $\bar{\mathbf{x}}-\mathbf{x}$ between the original and the deformed position.

Further, there are two types of forces $\mathbf{F}$ acting on the body — applied volume forces $\mathbf{f}$ acting in the volume of the body (as the gravitational force) and applied surface forces $\mathbf{g}$ acting on the boundary of the body — e. g. a traction or another object colliding with the body. As the response to the applied forces, internal forces appears in the volume of the body causing the deformation of the object [13]. In this paper, we are interested in the modelling of the static deformations when the applied and internal forces are in equilibrium state.

To establish the relation between the external entities, the theory of elasticity introduces internal entities covering the changes of the elementary volume element caused by the forces and displacement.

The internal entity corresponding to the displacement is *strain tensor*. There are several strain tensors used in the theory of elasticity, however our formulation is based on the non-linear Green-St.Venant strain tensor $\mathbf{E}$, which is defined as

$$\mathbf{E} = \nabla\mathbf{u} + \nabla\mathbf{u}^T + \nabla\mathbf{u}^T\nabla\mathbf{u} \tag{2.1}$$

The internal quantity associated with the forces is *stress tensor*. Our formulation uses the second Piola-Kirchhoff stress tensor $\sigma(\mathbf{x})$, which is related to the external forces by relation

$$(\mathbf{I} + \nabla\mathbf{u}(\mathbf{x})) \cdot \sigma(\mathbf{x}) = \mathbf{F} \tag{2.2}$$

where $\mathbf{I}$ is identity matrix and $\nabla\mathbf{u}$ denotes the gradient of the displacement vector [1].

2.2.2 Material Law and Constitutive Equation

The constitutive equation defines the relation between the applied forces and the displacement by coupling the strain and the stress tensors. Moreover, it determines the physical properties of the material.

We employ the hyper-elastic type of the material based on *stored energy function* $\mathbf{W}(\mathbf{E})$ [14] which couples the strain and stress tensor by:

$$\sigma(\mathbf{E}) = \frac{\partial\mathbf{W}(\mathbf{E})}{\partial\mathbf{E}} \tag{2.3}$$

The particular formulation of the function W depends on the chosen material law. So far, we have employed St.Venant material with the energy function defined as

$$\mathbf{W} = \frac{\lambda}{2}(tr\mathbf{E})^2 + \mu tr(\mathbf{E}^2) \tag{2.4}$$

where λ and μ are the *Lamé constants* from linear elasticity. However, we plan to employ some other non-linear material such as Mooney-Rivlin, formulated by the means of infinite series [14]:

$$\mathbf{W} = \sum_{rst=0}^{\infty} C_{rst}(\iota_1 - 3)^3(\iota_2 - 3)^s(\iota_3 - 3)^t, C_{000} = 0 \tag{2.5}$$

where $\iota_1 = tr\mathbf{E}$, $\iota_2 = \frac{1}{2}tr\mathbf{E}^2$ and $\iota_3 = |\mathbf{E}|$ are invariants of the strain tensor and C_01 and C_10 are material constants. This material has several non-linear extensions which fits well to numerous experimental data.

Putting together the relations 2.1, 2.2, 2.3 and 2.4 we get the governing partial differential equation (PDE) [15] which is non-linear due to the non-linearity of the strain tensor. The further non-linearity can in brought in by using some more complex material law.

2.3 Finite Element Method

Having the complex governing partial differential equations together with geometric description of the modelled domain, the finite element method provides an efficient tool for transforming these PDE into a large set of algebraic equations which can be solved by some numerical method.

The strategy of the method is to discretise the continuous domain of the deformed body by a mesh of elements with a simple geometry and compute the approximate solution in the nodes of the mesh. The advantage of the method is in providing the interpolation functions that can be used for approximating the solution over the volume of the elements [8, 16].

Based on a weak formulation of the input equation (an integral form combined with boundary conditions), the finite element method formulates the equations over particular elements of the mesh [17]. As this equation still contains the unknown function variable $\mathbf{u}(\mathbf{x})$, the approximation of the variable is performed by setting

$$U = \sum_i U_i \phi_i \tag{2.6}$$

where U_i is the discrete vector of unknown displacements in nodes of the mesh (also known as the *degrees of freedom*) and ϕ_i are the *interpolation functions*. Substituting the discretisation 2.6 into the element equations, we get a small system of algebraic equations for each element. After assembling together, one gets the large system of linear or non-linear equations depending on the input governing equation.

If the underlying PDE were linear, one arrives to a large sparse system of linear equations

$$\mathbf{Au} = \mathbf{f} \tag{2.7}$$

where the *stiffness matrix* $\mathbf{A}$ is assembled from the local element matrices obtained from the weak formulation of the governing PDE over the finite elements, $\mathbf{f}$ is the right-hand side vector computed from the force sources of the problem and $\mathbf{u}$ is the vector of deformation representing the displacements of the nodal points of the finite elements.

Moreover, if the governing problem of the PDE are non-linear, the finite element method gives a system of non-linear equations

$$\mathbf{A}(\mathbf{u}) = \mathbf{f} \tag{2.8}$$

where the coefficients of the non-linear matrix $\mathbf{A}(\mathbf{u})$ is obtained in the same way as in the linear case [18].

In case the governing equation contains a temporal derivative (e. g. when the viscoelasticity is assumed), the finite element method results in a system of ordinal differential equations, which can be solved by some well-known method (Runge-Kutta or Newark methods).

The size of the system is given by the number of the degrees of freedom, i. e. the number of the element nodes in the finite element mesh. The more accurate modelling is desired, the more fine-grained mesh must be generated and therefore the number of elements as well as the degrees of freedom is usually between thousands and millions.

To compute the solution i. e. the deformation vector **u**, several direct and iterative solvers of algebraic systems can be applied, depending on the internal structure of the matrices or the requirements on the speed and accuracy of the solution. As the stiffness matrices assembled from the element matrices are sparse and usually regularly structured, there are several approaches of parallelisation of both the direct and iterative solvers. The distribution of the matrix among the computational nodes is usually based on the decomposition of the domain being modelled. The techniques of the domain decompositions, direct and iterative solver parallelisation are presented in sections 3.2.1, 3.2.2 and 3.2.3, respectively.

Chapter 3

State of the Art

This section consists of two parts. First, we focus on several approaches in the are of the soft tissue modelling based on finite element method. Then we present a survey of the parallel solvers of the large linear and non-linear systems of algebraic equations which arise in finite element analyses.

3.1 Soft Tissue Modelling Based on FEM

The models of soft tissue can be classified by many criteria—static and dynamic, on-line and off-line computed, haptic interactive etc. We decided to classify the models as linear and non-linear, since we are mainly interested in the comparison of these two basic approaches.

3.1.1 Linear models with on-line computations

One of the first attempts of visual real-time soft tissue modelling is presented in the thesis by Bro-Nielsen in 1996 [14] and consequent articles [19, 20]. Using relations given by the theory of elasticity, the author derived linear finite element model of tissue deformation. In order to achieve visual real-time interaction with on-line computations, he applied the method of *condensation* simplifying the large linear system resulting from FE analysis. Since only the surface nodes are involved in the interaction, the global stiffness matrix K is condensed into smaller matrix K' including only the coefficients of surface nodes. The load and displacement vectors are condensed in the same way resulting in linear dense system $[K']u' = f'$ which is solved by direct or iterative solver. Adding mass and damping matrices with time-derivatives to the static model, the author obtained more realistic dynamic model. Euler semi-implicit integration scheme was used to discretise the time-dependent ODE. Using the dynamic model with 250 nodes, he achieved 20 frames per second rate. The method with condensation and several further optimisations is known as *Fast Finite Elements*.

Further, the author implemented an extension using parallelisation based on domain decomposition. The non-overlapping sub-domains are distributed among the processors, the shared nodes on the sub-domain boundaries constitute global system which is solved in the first phase. Then, the local systems containing internal nodes are solved using the preconditioned CG method. However, the parallel implementation using iterative methods showed to be too slow to achieve a real-time performance.

Similar approach using element-by-element method with on-line computations is described in [21, 22]. The main idea is not to assembly the global stiffness matrix but to solve the linear system representing internal forces for each element separately and then balance the forces in nodes connecting the elements. The model is dynamic with explicit time-integration, so the computation and balancing is performed in each time step. From the beginning the model is designed to be easily parallelisable—each element of FE model is assigned to a processor and one processor computes internal forces of several elements. If the force between two or several elements residing on different processors is balanced, the communication is established. Therefore, optimal distribution of elements among processors is of great importance.

The above described model was implemented in Laparoscopic Surgery Simulator **LASSO**. The aim was to achieve haptic real-time interaction using PHANToM haptic device and cluster of PCs performing the on-line computations. The prototype of the simulator is described in [23, 24].

In [25], a simple linear elastic model using condensation technique is described. The computations are performed online and the authors compare direct and iterative solution method. The iterative solution method achieves better results, however both are far from being applicable in haptic real-time interaction. The new idea presented in the article is multi-resolution method using a hierarchy of meshes. Employing the field programmable gate arrays provides another tool for speeding-up the computations and therefore, its combination with the proposed approach is of great interest.

3.1.2 Linear models with off-line precomputations

A new approach to soft tissue modelling was proposed by Cotin and Dellingette in [26] in INRIA project **Epidaure** [27]. They introduced linear static model allowing haptic interaction and involving some pre-computations. Adopting finite element method with mesh of tetrahedral elements they classified some nodes as fixed (cannot be displaced in space) and the others as free, and proposed *displacement-driven interaction* when the displacements of free nodes are imposed by a position of the haptic device and the action forces are computed according to the theory of elasticity. In order to speed up the interactivity rate, they took advantage of linearity and superposition principle [28]: at first for each free node k elementary displacement is set and (a) for each other free node n resulting displacement is computed and stored as tensor $[\mathbf{T}^{\mathbf{u}}_{\mathbf{nk}}]$ and (b) elementary force at node k is computed and stored as tensor $[\mathbf{T}^{\mathbf{f}}_{\mathbf{k}}]$. All tensors are computed by solving linear system using conjugated gradient method. During the interaction, the actual displacement induced by haptic device is expressed as superposition of elementary displacements, the mesh is deformed using tensors $[\mathbf{T}^{\mathbf{f}}_{\mathbf{ij}}]$ and the force feedback is computed using tensors $[\mathbf{T}^{\mathbf{f}}_{\mathbf{i}}]$. Using this method frequency varying between 300 and 500 Hz was achieved with model containing 8000 tetrahedra [29].

Since the previous model was static and due to the precomputations, any modifications of mesh topology (e. g. tearing or suturing) were not possible, in [30, 31] dynamic or *tensor-mass* model was proposed. In this case each tetrahedron is associated with data structure containing tensors corresponding to its vertices and edges. The tensors are computed on-line and determine the local stiffness between adjacent tetrahedra and therefore can be seen as some replacement of stiffness matrices. The local systems of differential equations are solved by explicit integration. If any modification of mesh topology occurs—some tetrahedra are removed and the local tensors in the adjacent tetrahedra are updated. Further details about cutting and tearing in this model can be found in [32, 33].

Since the model is computationally more expensive than the previous, *hybrid* models combining precomputation and tensor-mass approach were introduced. Despite the simplification, the rate of 25 time-steps per second achieved in experiments was too low for haptic interaction and therefore *force extrapolation* was used to increase the rate to 500 Hz. The details can be found in [34, 35].

The above described models developed in Epidaure project were successfully implemented in hepatic surgery simulator, using Alpha station for computations and Pentium station with **Laparoscopic Impulse Engine** for interaction [36].

Another static linear elasticity model with haptic interaction is presented in [37]. It is based on *small-area touch paradigm*, i. e. it assumes that the area touched by haptic device is small. Then, the global stiffness matrix K as well as its inverse K^{-1} are computed off-line and during the interaction, matrix A is extracted from the matrix K^{-1} and inverted. Since the size of matrix A is given by number of touched nodes, due to the small-area touch assumption the matrix A is small. The implementation of the model showed good results—running on Silicon Graphics Onyx2 with 8 CPU, the interaction with model consisting of 6,000 tetrahedra was performed.

The combination of precomputation and condensation is proposed in [38]. In the off-line mode, the global stiffness matrix K is assembled and further, the block $(K)_S$ with rows corresponding to the visible surface nodes is inverted obtaining $(K^{-1})_S$. By this simplification time as well as storage are being significantly reduced since K^{-1} is a large dense matrix. Mathematically, the elements of matrix $(K^{-1})_S$ describe how the influence of the unit force applied to a surface node propagates through the body to other surface nodes, and therefore the $(K^{-1})_S$ corresponds to the *Green's function* $G(x, y)$ which describes the propagation of the influence from point x to y.

3.1.3 Non-linear models

Picinbono ([39, 40]) extended *mass-tensor model* by including geometric non-linearity, i. e. using not linearised but complete Green-St.Venant strain tensor. Beside local stiffness tensors, he introduced global stiffness tensors, which are computed when creating the mesh for each vertex and edge. During the interaction the non-linear time dependent ODE is solved for each element by explicit integration scheme. The mesh topology changes caused by cutting or tearing are performed by removing the involved tetrahedra together with updating the stiffness data of adjacent elements. As an optimisation the model is treated as non-linear only if the current deformation exceeds a defined threshold. Otherwise, linear elasticity is preserved. Using the non-linear model with threshold, rate of 25 Hz was achieved [41] and force extrapolation was employed to enable haptic interaction.

The geometrically non-linear dynamic model of soft tissue with physically isotropic linear material is analysed in Zhuang's thesis [42]. Employing the explicit Newmark scheme the non-linear time-dependent equation with constant mass matrix M and damping matrix D is reduced to three linear systems which involves computation of matrix $M + \Delta t_n D$ where Δt_n is the n-th time step. The frequent solution of this problem is *mass lumping* i. e. diagonalisation of M and D. However, since mass lumping may cause loss of viscoelastic material properties, it is not applied. Instead, the length of time step Δt is restricted to a small set of values and for each value, the LU-factorisation of $M + \Delta t_n D$ is precomputed. Using the precomputed data in Newmark method, the visual real-time rate is achieved. More details about the method and results can be found in [43, 44].

Both geometrically and physically non-linear model is presented in [45]. It is based on concept of *dynamic progressive meshes* when a hierarchy of meshes together with refining and coarsening operators are generated off-line and during the interaction, the mesh is locally refined in the vicinity of place where the interaction takes place. The dynamic system is solved by explicit integration using mass-lumping. Modelling a tissue with 2,200 vertices, 20 frames per second is the achieved rate on Pentium III PC. The model has been implemented under project **VESTA** (Virtual Environment for Surgical Training and Augmentation).

The geometrical non-linearity is also assumed in model described in [46, 47]. Moreover, new dynamic material law is proposed including dependence of stress on both stress and its time-derivative which allows really dynamic effects as relaxation or creep. As simplification a material with *constant-Q* property is assumed, i. e. the ratio between stiffness and dumping remains constant [48]. The time equations are integrated by modified *Runge-Kutta* integration scheme which is also described in the article. The model again includes a hierarchy of meshes to be able to refine a part of mesh to reach higher level of detail. The results show much better dynamics properties of the proposed material, however do not include any notion about the length of computations and rate achieved in experiments.

3.1.4 Concluding remarks

In this section, we presented an overview of soft-tissue models. Apparently, the linear models are more frequently implemented than the non-linear, which are computationally much more expensive. Therefore, design of interactive models possessing geometrical as well as physical non-linear properties is of great interest.

The final note refers to topological modification of modelled objects. It is obvious that a realistic medical simulator must allow the user to perform the operations such as cutting and tearing. However, this puts another restriction on the model, mainly when precomputation is considered. Similarly, an interesting issue is simulation of suturing which is also the important part of surgical simulations. Some details and possible solutions can be found in [49, 50, 51].

3.2 Parallel FEM solvers

Since the complexity of problems modelled by FEM leads to discretisation resulting in a large number of the degrees of freedom (usually thousands or even millions for large and accurate models), the parallelisation of FEM analysis is of a great importance. The most expensive part of FEM analysis is solving the large linear or non-linear system of algebraic equations. On the other side, the great benefit of FEM is that the resulting system is sparse. This reduces the parallelisation of FEM to parallelisation of solver working with large sparse systems.

In the following we neither focus on particular physical problem nor examine the derivation of underlying FEM formulation. First we present short overview of domain decomposition techniques and next, we survey several efficient and frequently used parallel solvers.

3.2.1 Domain Decomposition

First, we would like to underline the difference between domain discretisation to finite elements introduced in section 2.3 and domain decomposition examined in the following text.

The term domain decomposition (DD) is widely used with two different meanings: in computer science it refers to the partitioning of computations among processors (*distributive DD*), whereas in mathematics it means the technique of subdivision of the domain where a boundary value problem is being solved (*numerical DD*) [52]. Since the domain is discretised by the finite element mesh, decomposition can be viewed as a subdivision of the generated mesh and therefore algorithms for partitioning of unstructured graphs and hyper-graphs can be applied.

In modern numerical analysis, the two definitions of DD are strongly interconnected, since the numerical DD provides very popular approach of constructing the iterative solvers, especially in multiprocessor environments [53].

The basic techniques are included in *one-level* DD, where the domain is broken to a set of overlapping or non-overlapping sub-domains. The two basic types of one-level sub-structuring are *Schur* and *Schwartz*.

In Schur method, the domain is divided to blocks (non-overlapping sub-domains), which are mapped to the processors in such way that no block can be split over two or more processors [54, 55]. The internal degrees of freedom (those which are inside the block) are eliminated by direct or iterative solver, resulting in a reduced system of equations containing only the interface DOFs (those on boundary of the block). The elimination is carried out on each processor independently and therefore in this phase no communication is needed. Finally, the much smaller reduced system is solved by iterative solver.

In Schwartz method, the domain is decomposed to sub-domains which can be both overlapping or non-overlapping depending on subsequent solver. The sub-domains are mapped to processors. The computation runs as follows: the problem is solved in each sub-domain, then the forces acting in the overlapped regions or across the sub-domain boundaries are balanced and then the sub-domain problem is solved again. This process is repeated until convergence is achieved [52].

To increase performance and address various convergence issues, *multilevel* DD techniques have been developed. As the name suggests, there is a hierarchy of several levels of decomposition. The bottom level provides the finest mesh which is being coarsened as climbing up the hierarchy. This approach is applied for design and implementation of multigrid solvers (see subsection 3.2.3).

Summary of domain decomposition methods together with further references can be found in [56]. The well-known program for mesh partitioning **METIS** and its parallel version **parMETIS** are both freely available on the Internet.

3.2.2 Direct Methods of Solving the Linear Systems

When speaking about direct solvers, we assume the linear system 2.7 introduced in section 2.3. Initially, the idea of direct solvers was based on factorisation of the stiffness matrix into lower and upper triangular matrix $[\mathbf{A}] = [\mathbf{L}][\mathbf{U}]$ and subsequent solution using Gauss elimination. However, the great disadvantage of this approach lies in huge storage requirements, since the assembly of the global matrix must be performed. Furthermore, the process of factorisation and direct solution is quite difficult to parallelise.

To cope with the above-mentioned issues, several new methods have been introduced. In [57] an approach for parallel direct solution based on the domain decomposition is proposed. The method involves ordering of the nodes in order to schedule the parallel computations as well as to minimise the number of the arithmetic operations. Another approach based on parallelisation of Cholesky factorisation was proposed in [58, 59]. The method was implemented in parallel solver **PSPASES** (Parallel SPArse Symmetric dirEct Solver).

A new direction in the development of direct solvers appeared by introducing *frontal method* in [60]. The method doesn't require factorisation $\mathbf{K} = \mathbf{LU}$, but assumes the matrix $\mathbf{K}$ to have the only non-zero elements in the rightmost column and in rectangular blocks $\mathbf{K_k}$ on the matrix diagonal. Instead of the whole matrix $\mathbf{K}$, only the blocks $\mathbf{K_k}$ are assembled into small dense *frontal matrix* which is solved much more easily. Since the method wasn't efficiently parallelisable, the *multi-frontal method* was introduced in [61]. In this case, there are several frontal matrices assembled by particular processors. Details and further improvements can be found in [62, 63, 64]. The methods have been implemented in the **HSL** library. The overview and comparisons with other direct solvers are available in [65, 66].

An interesting modification of the above method — *domain-based multi-frontal solver* was presented in [67]. The method can be regarded as the *recursive sub-structuring* technique. It was implemented in **IPSAP FEM** code which achieved high performance and good scalability results solving FEM problem with seven million degrees of freedom.

3.2.3 Iterative Methods for Linear Systems

First, we focus on the iterative methods for solving the linear system 2.7. These methods perform matrix-vector or vector-vector computations repeatedly until the convergence of solution is achieved. In some cases it may take too many iterations or even fail to find a solution. On the other hand, most of iterative solvers for systems resulting from finite element analyses is based on the domain decomposition giving a good possibility of parallelisation. That's the reason for iterative solvers to be preferred when analysing large scale physical problems.

There is a couple of techniques when solving linear systems iteratively. In the following we focus on Krylov methods and their modifications called FETI. Then we describe multigrid method and their applications.

Krylov methods The first class — Krylov subspace methods — provides an elegant means of designing iterative solvers that have proven to be very usable in practice [53]. A good description of mathematical background can be found in [68]. The main task concerning iterative methods is to speed up the computation, i. e. to decrease the number n of iterations. Since n depends on *condition number* which is given by ratio between maximal and minimal eigenvalue of matrix $\mathbf{K}$, various *preconditioners* have been developed to improve the condition of the matrix. A good overview of preconditioners together with further references can be found in [69].

The most known representative of Krylov technique is the *method of conjugated gradients (CG)* [70] designed for solving linear symmetric systems. Further improvements of the method as *preconditioned CG* (PCG), Bi-CGStab a Bi-CGStab2 has been proposed, see [71, 72].

There are two main approaches to parallelisation of Krylov method. The first approach is represented by *parallel normalised explicit preconditioned CG*. Since the explicit preconditioner is difficult to parallelise, this approach is not frequently used. The details of the method together with implementation and results can be found in [73].

The second approach is based on one-level domain decomposition or on *element-by-element* approach. When using one-level DD (see section 3.2.1), each processor iteratively solves one or more local systems corresponding to one or several sub-domains and the global matrix of the modelled system is never assembled. Since any balancing of forces across sub-domain boundaries constitutes communication among processors, much effort is exerted to minimise the communication by optimal decomposition of the domain (e. g. graph partitioning techniques are used [74]). The application of domain decomposition together with modified conjugated gradient vectors can be found in [75] (3D simulation of acoustic fields) and [76] (modelling of stress changes in rocks).

When using element-by-element methods, global matrix is never created and the matrices for each finite element are stored separately. Therefore, each processor solves several linear systems using iterative method. The examples of implementation can be found in [52] (magneto-hydrodynamics analysis), [77] (modelling of wind flow and rainfall), [23] and [21] (soft-tissue modelling).

Finite element tearing and interconnection

Another class of iterative solvers known as *Finite Element Tearing and Interconnection* (FETI) is some "extension" of the Krylov methods. The first FETI method can be described as two step Preconditioned CG algorithm — the sub-domain problems are solved in the first step and in the second the coarser problems of related sub-domains are examined and so the convergence is accelerated. To overcome problems with scalability, more efficient dual-primal FETI method was constructed ([78, 79].

The most significant implementation of FETI-DP method is represented by SALINAS — a software for high-performance structural and solid mechanics analysis. Today, SALINAS is treated as one of the most successful research codes for large scale parallel finite element analysis, since it is used for solving problems with more than hundred million of DOFs on ASCI Red and ASCI White computers [80, 81].

Multigrid Methods

The multigrid is a family of methods which fully utilise the multilevel domain decomposition [82] (see section 3.2.1). By grid we mean the mesh of nodes resulting from finite element analysis. In the hierarchy of grids, each node at the coarser grid level represents a set of nodes at the finer level and this coarsening is performed by *restriction operator.* We distinguish between algebraic and geometric multigrid by calling a method *algebraic multigrid* (AMG) if it uses only the values of matrix $\mathbf{K}$ (which represents the relations among finite elements 2.3) [83] in the construction of restriction operator. One such a method, *smoothed aggregation* and its parallelisation is described in [84, 85, 86] If the coarser grids are constructed explicitly using the geometry of the domain, the method is called *geometric multigrid.*

The multigrid solver runs as follows: having initial estimation $\mathbf{u_0}$, the residual $\mathbf{r} = \mathbf{f} - [\mathbf{K}]\mathbf{u_0}$ is computed and the grid hierarchy is traversed in V or W-cycles (the shape of letters symbolise ascending and descending the mesh hierarchy). On each level of the hierarchy an iterative solver (*smoother*) is called. The name comes from the fact that the residual $\mathbf{r}$ can be regarded as superposition of sine waves of different wavelengths and smoother reduces the components of error function with wavelength which is short with respect to the grid width. As the hierarchy is being climbed towards coarser grids, the errors with longer wavelengths are reduced. When the coarsest grid is reached, the system is small enough to be solved efficiently and fast using direct or iterative solver. Then, reverse process is performed—the solution is *prolonged* to the finer grids.

Since the coarsening and smoothening procedure can be performed on each sub-domain independently, multigrids show good parallelisation results. In [53] and [87] parallel AMG solver *Prometheus* built on parallel numerical library PETSc is introduced and described. The solver is successfully used in 3D modelling of human bones [88] with a system with several hundreds million DOFs. In [89] and [90] a parallel AMG solver for convection-dominated problems is described and recently, a parallel AMG algorithm using MPI was presented in [91] with application in computational fluid dynamics.

3.2.4 Iterative methods for non-linear systems

A general approach for solving non-linear problems 2.8 from section 2.3 consists in a successive approximation by a set of corresponding linearised problems. In each iteration of linearisation method such as *Newton* or *incremental loads method* (see [1]) linear system of equations is being solved. For example, in [75] the nonlinear acoustic field is modelled using a non-linear system of equations resulting from finite element analysis. The solution is performed by Newton method running BiCGStab linear solver in each iteration whereas in bone-tissue modelling, element-by-element preconditioned CG is called in Newton iterations [88].

Overall, the method of solving the non-linear system is given by three components — domain decomposition method, linearisation method and linear solver. This results in methods as *Newton-Krylov-Schwartz* where Schwartz domain decomposition is performed, the system is linearised by Newton method using Krylov linear solving in each iteration [53] or *Gauss-Newton-Krylov* used for computationally expensive inverse problem of wave propagation in [92].

3.2.5 Concluding remarks

In this chapter we focused on parallelisation of Finite Element Methods which is usually reduced to parallelisation of linear system solvers, since the standard analysis of non-linear as well as transient system results in large linear system of algebraic equations. Here, we shortly focus on different approaches to parallelisation of dynamic and non-linear solvers.

First, we focus on transient systems. In the standard semi-discrete approach a finite element mesh discretises space to generate a system of ODE in time that is then solved using time integration. When solving problems in electro-magnetics, *Finite Difference Time Domain* and *Finite Element Time Domain* methods are used. Comparison of parallelisations of the two methods can be found in [93]. Quite a new approach to transient systems has been proposed by *space-time meshing* method. In this case, the space-time is discretised using Galerkin finite element methods. Space-time meshing is suitable when solving PDE describing wave-like physical phenomena [94, 95]. In [96] *tent pitching algorithm* is proposed and tested on $1D \times time$ and $2D \times time$ non-linear problems.

Second, we would like to point out a new concept of *distributed point objects* presented in [97]. The parallel programming model is based on dynamic data structure addressed by points, it allows parallel refinement of objects and employs Krylov method together with multigrid preconditioner for solving the out-coming system.

Chapter 4

Objectives

In this section we present the basic objectives we have when designing the interactive model of the soft tissue simulation.

4.1 Geometric and Physical Non-linearity

The underlying relations of theory of the elasticity contain two basic types of non-linearities — the geometric and physical. Employing any type of non-linearities causes the underlying model to be non-linear and after applying the finite element method one arrives to a large non-linear system of algebraic equations.

The geometric non-linearity arises in definition of strain tensor that is the internal representation of the displacement. The strain tensor can be linearised by neglecting the non-linear terms, however this leads to unnatural behaviour of the soft body when large deformations occur and therefore only small deformations are modelled properly. Since the size of the deformations in our simulation should not be restricted and the behaviour should be realistic, we employ the non-linearised strain tensor in our model.

The physical non-linearity appears in relation between the internal entities, i. e. in *material law* coupling the stress and strain tensor. There are again simplified linearised material laws which do not employ the non-linear terms, however these do not fit properly to the experimental data measured in real soft tissue material. Therefore, we want to include the physical non-linearity to achieve the realistic behaviour of the simulation.

There are other types of non-linearities arising in definition of the boundary conditions. These non-linearities will be considered in case of more complex interaction between the modelled body and its environment.

4.2 Haptic Interaction

Besides the visualisation, the haptic interaction is the most important part of the soft tissue simulation. The user is equipped with a haptic device, which is coupled with *haptic interaction point* that can be represented by some virtual probe in the scene (stylus, scalpel etc.). By moving the probe the user can touch the other virtual deformable objects.

The interaction is enclosed in the *haptic loop*:

– the user places the virtual probe at some position

– the collision detection is performed and if the probe interacts with a body, the deformation of the body together with the reaction force are computed

– the force is delivered to the user via the feedback of the haptic device

The main issue of the haptic interaction is the refresh rate of the haptic loop. To simulate the smooth motion of the haptic device, when colliding with some virtual object, the necessary refresh rate is above 1 kHz. Therefore, the time for computation of the deformation and reaction forces is shorter than 1 ms. In case of non-linear model, such a computation is not feasible and therefore our objective is to propose a pre-computation scheme, allowing the combination of the non-linear model and haptic interaction.

4.3 Configuration Space Precomputation

One possibility how to cope with the issue described in the previous section is to divide the interaction into two phases — first, the configuration space is precomputed in the off-line phase and than, the precomputed data are used during the interaction.

In the state-of-the-art section 3.1, we presented some approaches of the pre-computation when the model is linear (*elementary displacements*) or non-linear (*mass tensors*). However, these approaches are not suitable for achieving the fully realistic non-linear behaviour. Therefore, our objective is to propose a novel approach for precomputation, when a large set of configurations is precomputed. Here, by the configuration we mean the particular position of the probe together with the corresponding deformation. Since it is not possible to compute all the configurations of the model, we propose a discretisation of the configuration space of the soft body.

In the second phase, the interaction is performed using the precomputed configurations. Since these are computed only in the discrete points, the interpolation process is applied to obtain deformation and forces for any position of the probe in the space of the soft body. Our objective is to find and implement suitable interpolation functions that would be easily computable on a single computer within the haptic loop.

4.4 Topological Changes and Adaptive Remeshing

The main application of the soft tissue modelling is in medical training and operation planning. In both cases, the actions as cutting, tearing and burning the soft tissue are of the paramount interest. Therefore, it is necessary to allow the user to perform these operations within the simulation. However, the underlying physics of such processes is far from being understood and the proper physical simulation is not so far possible.

The most known workaround of this profits from the fact that the continuum body is modelled as a three-dimensional mesh of finite elements and the above mentioned processes are then performed as the topological changes of the mesh. There are two basic approaches — first the cut is performed by separation of neighbouring elements. However, this approach requires an alignment of the boundaries of the relevant elements. Another possibility is the removal of the elements lying in the area of the cut.

For both approaches the size of elements which are separated or removed must be very small, however this enormously increases the number of variables in the model in case of large bodies. Therefore, the concept of the adaptive re-meshing is suitable to address this issue — the mesh is adaptively refined in the area actually close to the virtual probe, whereas the other parts of the mesh are coarsened.

Our objective is to combine this approach with the precomputation scheme, so the topological changes could be performed efficiently only at the cost of increased size of the configuration space.

4.5 Speed-up of the Precomputations

The precomputation phase includes a large number of configuration computations (from hundreds to dozens of thousands in case of adaptive re-meshing). Moreover, each computation consists of assembling and solving the large system of algebraic equations. Therefore, this phase is computationally expensive and not suitable for sequential computations.

Therefore, we want to focus on the parallelisation of the precomputation phase. More precisely, there are two levels of possible parallelisation — first, the overall computation of the configuration space can be performed in a distributed manner and second, the particular methods of the numerical solution can be parallelised.

Focusing on the first level, there are several approaches for parallel state-space search. However, besides the search, we also need a full reconstruction of the space and therefore we employ a modified version of Transposition-Driven Table Scheduling algorithm [98], which allows distributed and efficient traversing of the configuration space and storage of the discovered states so the space can be later fully reconstructed.

In the case of the second level of the parallelisation, we would like to employ some of the algorithms for parallel solving the large systems arising in the finite element analysis. The iterative multigrid methods will be of interest since they correspond with the idea of the adaptive meshing, however the other techniques as parallel direct solvers could bring interesting results as well.

Our objective is to find an efficient combination of the two approaches of the parallelisation described above. First, we focus on the first type of the parallelism, which is suitable for this type of computations. The second type will be considered mainly in connection with the extension sketched in the next section.

4.6 Online Pre-computation

After successful implementation of the precomputation scheme, we would like to experimentally focus on a modification of this approach, which would "push" the precomputation phase towards the interaction phase, as the precomputation would be performed during the interaction phase (therefore the name *on-line precomputation*).

This approach is based on prediction of the further motion of the virtual probe and the configurations are precomputed in the discrete steps along the possible paths where the probe could move in next moments. This means that the portion of the state space which could be needed in the interaction is precomputed in advance and in the moment, when the probe reaches some position, the corresponding configuration (the deformation and reacting force) can be computed by the fast interpolation process from the data just precomputed.

Employing this method, the expensive over-all precomputation wouldn't be applied and therefore much larger state space could be still feasible. The main advantage here is the possibility of employing the dynamics in the model, which is again necessary for improving the realistic behaviour of the model. However, this approach requires not only huge computational resources used directly during the interaction, but also low-latency and reliable network connection among the computing nodes and the machine running the haptic loop.

Chapter 5

Proposed Approaches

In this section, we describe the approach that we propose and apply in our interaction model. First, we focus on the formulation of the elasticity problem, how to extend the formulation in order to employ the haptics and finally we present the precomputation scheme together with some further optional improvements.

5.1 Mathematical Model and FEM Formulation

The mathematical model is based on the theory of the elasticity, more precisely on the relations introduced in section 2.2. As we want to model the large deformations, we use the non-linear strain tensor 2.1. Further, the equilibrium relation 2.2 is employed to relate the stress tensor with the applied forces. The hyper-elastic type of the material 2.3 is considered, with the energy function defined according to St.Venant 2.4 or 2.5 material law.

The mathematical model is subjected to the finite element method described in section 2.3. Since the relations are non-linear, the FE method results in a system of non-linear equations

$$\mathbf{A}(\mathbf{u}) = \mathbf{f} \tag{5.1}$$

where $\mathbf{A}(\mathbf{u})$ is the stiffness matrix defined as

$$\mathbf{A}(\mathbf{u}) = \int_{\Omega^e} \frac{\partial W}{\partial E_{jk}} \left(\delta_{ij} + \frac{\partial u_i}{\partial x_j} \right) \frac{\partial w_i}{\partial x_k} dx \tag{5.2}$$

and $\mathbf{f}$ is the load vector

$$\mathbf{f} = \int_{\Omega^e} fw dx + \int_{\partial\Omega^e} gw da. \tag{5.3}$$

The exact derivation of the above relations can be found in [99].

So far, we have implemented the model based on the St.Venant material law using the above finite element formulations. The implementation was performed in Matlab environment using Tensor toolbox. The deformations of three-dimensional object were modelled, using the finite element mesh generated by open-source program `gmesh`. Some simple non-homogeneity was included using different Lamé coefficients for different parts of the body (see the Fig. 5.1).

We plan to experiment with some more complicated material types, including non-homogeneous or viscous properties. If time permits, the contact problem concerning two soft bodies will be considered and analysed as well.

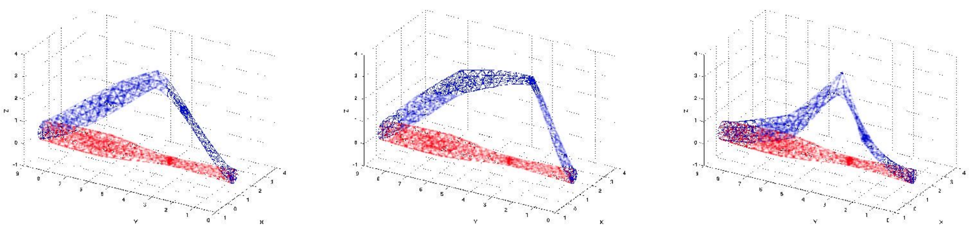

Figure 5.1: The deformation of a virtual object. One point of the surface is displaced and the resulting deformation of the entire body is computed using the experimental implementation. The virtual object is divided into three parts of the same length. In the left figure, the stiffness is the same for the entire body. In the middle figure, the central part is much rougher then the sides, whereas in the right figure, the central part is softer than the rest of the body.

5.2 Displacement-driven Interaction

As introduced in section 2.2.1, the standard deformation modelling helps us to formulate and solve the problem of computing a deformation (more accurately the displacement) of a body after the forces have been applied. However, the situation in haptics is slightly different. Controlling a haptic device we interact with the soft body by touching its surface by some probe (a virtual representation of the haptic interaction point) displacing a part of the surface and we need to compute the reacting force and the entire deformation. This approach is usually referenced as the *displacement driven interaction.*

Further we expect the FEM discretisation to be already computed and by interacting with the virtual body we mean the interaction with its final element mesh. Then the interaction works as follows:

- we move the virtual probe towards the modelled body
- for the actual position of the probe, we perform collision detection between the probe and the FE mesh, if there is some, we get the displacement of the nodes *directly touched* by the probe
- having the set of the displaced nodes, we compute the deformation of the entire body as well as the reaction force acting on the probe

To cope with this scheme, we apply *Lagrange multipliers* approach on the system of non-linear equations $\mathbf{A}(\mathbf{u}) = \mathbf{f}$ derived is section. For the sake of simplicity we suppose there is only the i-th node touched by the probe and its displacement is $u_i = t$ known as *pseudo-load.* The force f_i exerted in this point is unknown and we denote it by a variable h. This means the i-th equation of the non-linear system is modified as:

$$\mathbf{A}_i(u_1, \dots u_n) = h \tag{5.4}$$

After simple operation we have a modified equation having $n+1$ variables:

$$\hat{\mathbf{A}}_i(u_1, \dots u_n, h) = \mathbf{A}_i(u_1, \dots u_n) - h = 0 \tag{5.5}$$

Since we have added a new variable, we have to add a new equation to the original system $\mathbf{A}$. This is simple, since we know the prescribed displacement t for the i-th node. Therefore the new equation is:

$$\hat{\mathbf{A}}_{n+1}(u_1, \dots u_n, h) = u_i = t \tag{5.6}$$

If there are prescribed displacements (the pseudo-loads) of $k > 1$ nodes, the technique is analogical:

- the vector of variables $\mathbf{u}$ is augmented by the unknown forces $\mathbf{h}$ (denoting $\mathbf{u}|\mathbf{h}$)
- the right hand-side vector of known forces $\mathbf{f}$ is augmented by the pseudo-loads $\mathbf{t}$
- the equations corresponding to the touched nodes are modified analogically as described above
- k new equations are added to the original system A putting the displacements of the touched nodes to the prescribed values

After applying the modifications we get the system

$$\hat{\mathbf{A}}(\mathbf{u}|\mathbf{h}) = \mathbf{f}|\mathbf{t}. \tag{5.7}$$

So far, we experimentally demonstrated the approach through two implementations of the Lagrange multipliers. First, we created a model of two-dimensional thin plate using linear elasticity described in [8]. The user could deform the plate with a small spherical probe controlled via the Phantom haptic device. The vector of the prescribed displacements was computed using the simple collision detection between the probe and the plate, smoothened by the interpolation functions from the underlying finite element method. Since the model was linear and *small area paradigm* [37] was applied, the computations were performed on-line in haptic loop allowing meshing by 625 elements. This implementation helped as to better understand the finite element method and and modelling of simple deformable objects for the haptic interaction.

Another implementation of the approach was performed using the non-linear model in Matlab. For the sake of simplicity, the vector of the prescribed displacements was obtained by pushing a surface node of the finite element mesh to some position. Then, the extended non-linear system 5.7 was assembled and solved. Due to this experimental implementation of the non-linear model, we became more familiar with the tensor formulation of the problems in non-linear elasticity. Moreover, it will be included as the core of the models implemented in future.

In the first phase, we plan to employ some fast collision detection algorithm, computing the vector of the prescribed displacements. The algorithm should be able to handle more complicated geometry of the soft body as well as the probe representing some surgical instrument. However, any such collision detection is based only on the geometry of the soft body. This means, that the local deformation of the tissue in the close vicinity of the surgical tool is not modelled correctly according to the physical properties of the model. This could be solved by either including some minimisation of an energy potential representing the penetration of the objects directly into the FEM, or considering the interaction as a contact problem, where both the soft body and the tool are modelled physically. However, both approaches enforce a non-trivial modification of the mathematical model and can lead to much more complicated formulations.

5.3 Numerical Solution of the Non-linear System

In this section, we use the notation $\mathbf{A}(\mathbf{u}) = \mathbf{f}$ for the system being solved, nevertheless in our model and its implementation, the solution method is applied to the extended system 5.7.

The system can be solved using an iterative method. Generally, in each iteration, having the solution estimation $\mathbf{u}^{(n)}$, the new estimation is computed as

$$\mathbf{u}^{(n+1)} = \mathbf{u}^{(n)} + \delta\mathbf{u}^{(n+1)} \tag{5.8}$$

The displacement increment $\delta\mathbf{u}^{(n)}$ is computed from linearised system depending on the method being used. In *method of incremental loads*, we let the body force vary by a small force increment

$$\delta\mathbf{f}^{(n)} = (\lambda^{(n+1)} - \lambda^{(n)})\mathbf{f}, \qquad 0 \geq \lambda^{(n)} \geq 1 \tag{5.9}$$

and then, we compute the displacement increment from linear system

$$\mathbf{A}'(\mathbf{u}^{(n)})\delta\mathbf{u}^{(n+1)} = \delta\mathbf{f}^{(n)} \tag{5.10}$$

where $\mathbf{A}'(\mathbf{u})$ is *Fréchet* derivative or tangent stiffness matrix that is defined as derivative of the stiffness matrix with respect to the variables. Since the behaviour of the body being modelled can be viewed as a *response* of the system to the applied forces and pseudo-loads, the incremental method constitutes the *response path* of the system which mirrors the behaviour of the body when increasing the forces.

In *Newton* method, the linearised system is of the form

$$\mathbf{A}'(\mathbf{u}^{(n)})\delta\mathbf{u}^{(n)} = \mathbf{f} - \mathbf{A}(\mathbf{u}^{(n)}) \tag{5.11}$$

where $\mathbf{A}(\mathbf{u}^{(n)})$ is defined by relation 5.2.

Both methods can be used simultaneously — the method of incremental loads is used as a predictor and the Newton method as the corrector as follows: we start the computation with zero force vector and we increase the force in successive steps computing the displacement by the incremental load method. After several steps (depending on the size of residual), we perform correction step by using Newton method taking the actual displacement vector as the initial estimation. This process is iteratively repeated until the desired force vector is achieved and the corresponding displacement computed with the requested residual.

We applied the predictor — corrector approach in the solution of the non-linear system implemented in the Matlab environment. Having some target position of a selected surface node of the finite element mesh, we divided the response path between the rest and target position of the node into n steps. Then we incrementally increased the displacement of the selected node and for each step, we solved the linearised system 5.10 using the deformation and force estimations from the previous iteration. After the n-th increment, we performed the correction phase using the Newton method with the result from the last prediction step as the initial estimation. We compared the results with the solution, obtained by the Newton method 5.11 starting with zero estimation. It turned out, that the results were almost identical. Moreover, in the case of the predictor — corrector scheme, we had quite good estimations of the deformation in the intermediate states.

Further, we would like to focus on this scheme in case of more complicated deformations, including some non-trivial response paths, when plastic irreversible deformations occur. We believe that the prediction part of the method can help us to follow the correct response path, if the correction phase is applied sufficiently often. Therefore, new criteria for correction application will be studied as well.

5.4 Off-line Precomputation Scheme

In the previous section, we presented deformation modelling based on finite element method and described the slight modification due to the different input in the case of the haptic interaction. However, the section concerning with the numerical solutions shows, that in the case of the non-linear system, the tangent stiffness matrix is assembled and either the linearised system 5.10 is solved or even the more expensive correction 5.11 is applied in each step. This cannot be performed in each iteration of the haptic loop even on the most powerful computers, since not only the power, but also the latency of the computations is of the great importance.

Therefore in this section, we present our strategy for solving this problem, based on off-line precomputation of a finite configuration space of the interaction. First, we focus on the idea of the precomputation algorithm, then we get to the interaction phase based on the interpolation of the precomputed data.

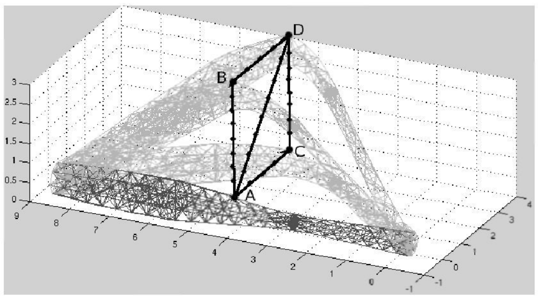

Figure 5.2: Four configurations of the system with deformable body — starting in the configuration A, a point on the surface of the body was displaced to positions B,C and D

5.4.1 Precomputation Algorithm

Since only a finite configuration space can be efficiently precomputed, we introduce a new discretisation of the space, where the soft body is placed. To distinguish this discretisation from the one used in finite element method, we call this *space discretisation.* It is performed by creating tree-dimensional regular grid covering the entire space where the soft body is placed. The grid consists of nodal points called *check-points* and paths between the adjacent check-points.

The first idea is to put the probe into each check-point of the 3-D grid and to compute the configuration for this position. However, this is not sufficient, as the actual configuration depends not only on the actual position of the probe, but on the history of the travelling as well, e. g. if the soft body is represented by a vertical plate perpendicular to the x axis, the deformation is different when hitting the plate by the probe from the left and from the right. Therefore, instead of instant placing the probe into a check-point, we compute the transitions between adjacent check-points, starting in a check-point outside the soft body. In each check-point we establish *levels of configurations* and each time we traverse the path from the source check-point to the target, we compare the actual configuration with those already stored in the target. If it differs from each of them by some given factor, we add a new level and store the configuration there. In addition, we also store the number of the level reached in the target check-point to a data structure in the source check-point.

Not all the configurations of the body are reachable in practice, since the feedback force applied to the haptic device is restricted to some interval due to mechanical construction of the device. This fact can be used to establish some bound on the precomputation of the space as follows: if the value of the force of a new computed configuration lies inside the restricted interval, it is stored in a new level which is marked for further expansion. This means the configuration will be later used as a source state for computing the transitions to the adjacent check-points. Otherwise, the configuration is stored as well, however it is not expanded anymore.

As the number of transitions in the finite grid is bounded and the number of the levels in each check-point is restricted as described above, the precomputed space is finite and the computations eventually terminate.

So far, we have experimentally implemented the precomputation algorithm in the Matlab environment. Only a small portion of the finite configuration space was computed. More precisely, 13 states were computed together with 13 paths between them. As proposed in section 5.3, the paths were traversed by the prediction steps and in each state, the correction was applied. In the Fig. 5.2, there are four configurations of the modelled object together with the path between them. First, we want to complete the implementation, so it generates the entire space using some efficient way of handling the configuration levels. Then, we want to focus on the adaptive re-meshing, when the existing elements are either divided into smaller parts or they are reconnected to decrease the number of FE nodes. The question is, whether the re-meshing can occur during the prediction phase or must be accompanied also by the correction.

Another issue that will be analysed is the modification of the topology — the non-physical modelling of the operations as cutting or tearing. The adaptive meshing is the first step to the successful realisation of this extension, however the possibility of changing the topology of the modelled object can lead to a huge increase of the state space size. We will experimentally analyse this extension together with the parallelisation of the computations (see section 5.5).

5.4.2 Interpolation Phase

During the interaction, the deformation and reaction force for any position of the probe are computed on-line by interpolation of the precomputed values. For the actual position of the probe, the set of the closest configurations is chosen. The motion of the probe in space can be viewed as following a path, which is interpolated by the discrete virtual paths computed and stored during the precomputation phase.

The interaction consists of two basic parts — the user haptically manipulates the virtual probe feeling the reacting force and the entire soft body is visualised on a display or stereo projection. Therefore, the interpolation consists of two threads — the first operating the haptic device and the second performing the visualisation.

The haptic thread runs in real-time mode with a high refresh rate. It computes the interpolation that must be smooth and precise enough so that the haptic part of the model behaves realistically. As we cope with the interpolation of three numbers (the force vector), we can afford to employ the interpolation functions of high orders getting better results still within the haptic loop.

The visualisation of the deformation is more computationally expensive, since all the degrees of freedom of the soft body must be interpolated. However, the refresh rate of the visualisation gives us enough time to do this and as the accuracy of the visual perception is not so high as in the case of haptics, we can use the interpolation functions of lower order.

Using the experimental implementation, we precomputed a part of the configuration space as sketched in the previous section 5.4.1. Then we simulated a motion of the probe along a random path $\mathcal{P}$ through the precomputed portion of the space and we computed the deformation and force vector along the path employing the linear interpolation of the precomputed states enclosing the path.

For the purpose of comparison, we performed an exact computation of the random path $\mathcal{P}$. We selected a point on the surface of the soft body and pulled it along the path $\mathcal{P}$ in small steps. In each step, we computed the exact deformation and force by the Newton method. The relative error between the interpolation-based and the precise method is depicted in Fig. 5.3.

We observed that albeit simple linear interpolation has been employed, the interpolated values are close enough to those computed by the precise method along the entire path. Taking the maximal force computed by the precise method as the base, the differences between the values computed by the two methods did not exceed 4% and the relative deformation error computed as the normed difference between the displacement vectors remains under 14%. This confirms the validity of the proposed algorithm.

In future, we will focus on interpolations of a higher order, since in case of really non-linear complex response path, the linear interpolation would not work reasonably. Also, in case of complex configuration space including many levels of configurations, e. g. due to the adaptive re-meshing and topology changes, the selection of the enclosing configurations of the actual path is not so trivial. In this case, the selection will be again determined by the history of the interaction, and the process can be viewed as *finding a tunnel* around the actual interaction path through the

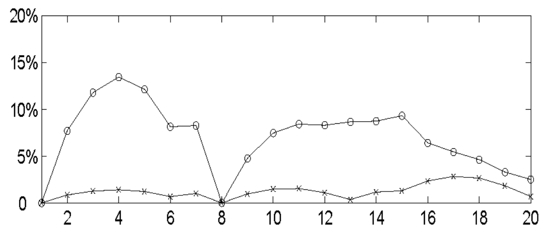

Figure 5.3: The relative force and deformation error of the interpolated method against the precise Newton computations of 20 steps of a random motion of the probe

discrete precomputed space. It is clear, that due to the short latencies which are necessary for a stable interaction, the tunnel must be constructed in advance, i. e. when the probe arrives in some particular position, the enclosing precomputed configurations have been already determined and only the fast interpolation is performed. However, this can lead to another algorithm, which makes an on-the-fly prediction of possible future directions of the probe. We will therefore analyse the usability of such approach.

5.5 Parallelisation of the Precomputations

To accelerate the precomputation phase, we employ some methods of parallelisation. There are two possible levels:

Lower-level parallelism. This parallelisation is applied in the computation of a particular path between two configurations. First, the construction of the tangent stiffness matrix can be done easily in parallel, since the submatrices corresponding to distinct elements can be computed independently. Then, the solution of the linearised system in the prediction or correction phase can be solved by parallel solver. We will employ some of the direct and iterative solvers mentioned in the section 3.2. The main advantage of the direct solver is that the number of operations can be predicted in advance. On the other side, the iterative solvers based on the multigrid method reflects the hierarchy of finite element meshes that can be successfully applied in the case of the adaptive meshing.

Higher-level parallelism. In this case, state space search is performed in distributed manner, e. g. the independent configurations are computed at once in different parts of the distributed environment. Here, we would like to employ the TDS algorithm [100], which allows the distributed state space search, together with its modification [98].

The main issue being addressed will be how to combine the both approaches presented above to obtain the efficient parallelisation of the computations. We think, that in the case of the off-line precomputation of a large configuration space, the higher-level parallelism should be sufficient, since in that case the TDS algorithm provides a good way of distributed processing with minimal overhead. In case both approaches are used in combination, the computation of each state is performed on the set of nodes, instead of a single node. Therefore, a load-balancing among such sets of nodes becomes of great interest, however for now, we don't find addressing this issue so important.

5.6 On-line precomputation scheme

The idea of the process of finding a tunnel presented in the last part of the section 5.4.2 lead us to a modification of the overall precomputation scheme. Provided we have sufficient computational resources, we can replace the *finding a tunnel* by *constructing a tunnel.* This means that we perform the on-line computation of the configurations, which are going to be needed for the interpolation of the force and deformation vector for the probe positions in the on-coming moments.

In the case that the enclosing configurations are computed, i. e. the actual part of tunnel is constructed, the forces and deformations for all the positions of the probe inside this part of the tunnel can be easily computed by the interpolation. This means, that the computation of the tunnel can be run in a loop on much lower frequency than the haptic loop. On the other hand, for the current part of the tunnel, all possible directions must be computed, so if the probe chooses some direction, the correct elongation of the actual tunnel is used for further interpolating. Moreover, the order of the interpolation determines the number of configurations, which must be precomputed. As we expect some higher-order interpolation will be needed, the number of the pre-computed configurations can be significantly increased (in order of dozens). This implies, that such on-line computations demand large resources and all possibilities of the near-future motions must be computed in a very short time. There are several possibilities, how to do this.

In the first place, the parallelisation of the computations can be exploited, mainly the lower-level type, since the number of the configurations is not so large, whereas each transition must be computed as fast as possible. We think, that this type of application is suitable for smaller clusters consisting of powerful nodes (possessing several cores) connected by some low-latency network such as Infiniband. The architecture of such a system would be based on a central computational node, which must interconnect the interface of the haptic device (usually some Legacy ISA bus) on one side and the interface of the low-latency network (usually represented by a PCI-Express card) on the other. Possibly, the effective interconnection of the two different interfaces will involve virtualisation, if each interface is controlled by a different operating system. Therefore, this issue will be addressed in cooperation with researchers dealing with the application of the virtual machines.

Further, some advanced technology such as *field programmable gate array* (FPGA) can be employed. There are several issues which must be addressed. First, the FPGA can be successfully applied in case when the type of the algorithm is suitable for the computational model provided by the gate arrays. Mainly, the data exchange and the communication between the master processor and FPGA seriously reduce the performance and therefore, the FPGA should be very close to the haptic loop, e. g. directly computing the actual force for the haptic device. Further, the algorithm should fit into the FPGA, since its reconfiguration is too expensive to be performed inside the haptic loop frequently. On the other hand, the reconfiguration could be applicable in case of the topological changes, since the operations as cutting or tearing take some time, as the surgical tool has to overpower the resistance of the tissue. This would allow the master processor to reconfigure the gate array in order to adapt efficiently to the new topology of the body being modelled. Besides FPGAs, there are also some other technologies providing similar properties, e. g. cell processors.

It is clear, that this approach of the on-line precomputation brings many advantages. First, the user does not have to wait for the precomputations of the entire space. Further, since the entire space is never constructed, the topological changes and irreversible deformations can be managed much easier. Moreover, it allows us to include the dynamic properties of the system, since we believe that the on-line precomputations can be combined with explicit integration.

However, we are not able to say, if such computations are sustainable during the interaction due to the high sensitivity of the haptic device on the one side, and the long latency and possible failures of the computational nodes and interconnecting network on the other. This will be subject of further long term research.

Chapter 6

Proposed Plan of Work

- **Spring 2007**
  - Full implementation of the off-line precomputation phase
  - Experimental comparison of several types of the collision detection between the soft body and the probe, modelling some more complicated deformations
  - Haptic interaction and testing the interpolation methods (submission of paper with results)
  - Higher-level parallelisation of the precomputation phase
  - Inclusion of the adaptive re-meshing and topology changes (submission of paper with results)
- **Autumn 2007**
  - Comparison of various types of material laws and boundary conditions
  - Experimental implementation of the parallel on-line precomputation approach with static model, including a prototype of hardware and software architecture interconnecting the haptic device and low-latency network
  - Extension of the model by dynamics and more complex operation modifying the topology (submission of paper with results)
  - Studying some further improvements as the contact problems modelling an interaction of two soft bodies
- **Spring 2008**
  - Submission of a journal article presenting the overall results
  - Finishing the thesis
  - Thesis submission an the end of March 2008

Bibliography

[1] E. Gladilin. *Biomechanical Modeling of Soft Tissue and Facial Expressions for Craniofacial Surgery Planning.* PhD thesis, FU Berlin, Germany, 2003.

[2] F.L.Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 567–585, 1989.

[3] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 151–160, New York, NY, USA, 1986. ACM Press.

[4] S. Coquillart. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 187–196, New York, NY, USA, 1990. ACM Press.

[5] S. Cover, N. Ezquerra, J. O'Brien, R. Rowe, T. Gadacz, and E. Palm. Interactively deformable models for surgery simulation. *IEEE Comput. Graph. Appl.*, 13(6):68–75, 1993.

[6] M. A. ElHelw, A. J. Chung, A. Darzi, and G.-Z. Yang. Image-based modelling of soft tissue deformation. In *Proc. MICCAI 2003*, 2003.

[7] D. L. James and D. K. Pai. Artdefo—accurate real time deformable objects. In Alyn Rockwood, editor, *Siggraph 1999, Computer Graphics Proceedings*, pages 65–72, Los Angeles, 1999. Addison Wesley Longman.

[8] J.N.Reddy. *An Introduction to the Finite Element Method.* McGraw-Hill, 1993.

[9] D. James and D. Pai. A unified treatment of elastostatic contact simulation for real time haptics. In Haptics-e, The Electronic Journal of Haptics Research, September 2001, Vol. 2, No. 1, 2001.

[10] D. James and D. Pai. Real time simulation of multizone elastokinematic models. In *International Conference on Robotics and Automation*, pages 927–932, Washington, D.C., USA, 2002.

[11] C. Basdogan, S. De, Jung Kim, M. Muniyandi, Hyun Kim, and M. A. Srinivasan. Haptics in minimally invasive surgical simulation and training. *IEEE Comput. Graph. Appl.*, 24(2):56–64, 2004.

[12] R. Balaniuk and K. Salisbury. Dynamic simulation of deformable objects using the long elements method. In *Proceedings on the 10th Symp. On Haptic Interfaces for Virtual Env. & Teleoperator systems (HAPTICS '02)*, 2002.

[13] J. H. Heinbockel. *Introduction to Tensor Calculus and Continuum Mechanics.* Trafford Publishing, 2001.

[14] M. Bro-Nielsen. *Medical Image Registration and Surgery Simulation.* PhD thesis, IMM Technical University of Denmark, 1996.

[15] W. A. Strauss. *Partial Differential Equations, An Introduction.* John Wiley & Sons, Inc., 1992.

[16] C. A. Felippa. Introduction to finite element methods. Under review by John Wiley & Sons. Inc, Material for Finite Element Courses, http://www.devdept.com/documentation.html.

[17] C. A. Felippa. Advanced linear finite element methods. Under review by John Wiley & Sons. Inc, Material for Finite Element Courses, http://www.devdept.com/documentation.html.

[18] C. A. Felippa. Geometrically nonlinear finite element methods. Under review by John Wiley & Sons. Inc, Material for Finite Element Courses, http://www.devdept.com/documentation.html.

[19] M. Bro-Nielsen and S. Cotin. Real-time volumetric deformable models for surgery simulation using finite elements and condensation. *Computer Graphics Forum*, 15(3):57–66, 1996.

[20] M. Bro-Nielsen. Fast finite elements for surgery simulation. In *Proc. Medicine Meets Virtual Reality 5*, 1997. 6 pages.

[21] A. C. Rhomberg. *Real-Time Finite Elements: A Parallel Computer Application.* PhD thesis, Swiss Federal Institute of Technology, Zurich, 2001.

[22] G. Székely, Ch. Brechbühler, R. Hutter, A. Rhomberg, and P. Schmid. Modelling of soft tissue deformation for laparoscopic surgery simulation. *Medical Image Analysis*, pages 57–66, March 2000.

[23] G. Székely, M. Bajka, and C. Brechbühler. Virtual reality based surgery simulation for endoscopic gynaecology. In *Proceedings of Medicine Meets Virtual Reality 7*, pages 351–357, Amsterdam, 1999. IOS Press.

[24] G. Székely, C. Brechbühler, and J. Dual et al. Virtual reality-based simulation of endoscopic surgery. *Presence: Teleoperators and Virtual Environments*, 9(3):310–313, June 2000.

[25] A. O. Frank, I. A.Twombly, T. J. Barth, and J. D. Smith. Finite element methods for real-time haptic feedback of soft-tissue models in virtual reality simulators. In *VR '01: Proceedings of the Virtual Reality 2001 Conference (VR'01)*, page 257, Washington, DC, USA, 2001. IEEE Computer Society.

[26] S. Cotin, H. Delingette, and N. Ayache. Real time volumetric deformable models for surgery simulation. In *VBC*, pages 535–540, 1996.

[27] N. Ayache. Epidaure: a Research Project in Medical Image Analysis, Simulation and Robotics at INRIA. *IEEE Trans. on Medical Imaging*, 22(10):1185–1201, October 2003.

[28] S. Cotin, H. Delingette, and N. Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions On Visualization and Computer Graphics*, 5(1):62–73, January-March 1999.

[29] N. Ayache, S. Cotin, H. Delingette, J.-M. Clément, J. Marescaux, and M. Nord. Simulation of endoscopic surgery. *Journal of Minimally Invasive Therapy and Allied Technologies (MITAT)*, 7(2):71–77, July 1998.

[30] S. Cotin, H. Delingette, and N. Ayache. A hybrid elastic model allowing real-time cutting, deformations and force-feedback for surgery training and simulation. *The Visual Computer*, 16(8):437–452, 2000.

[31] G. Picinbono, J-C. Lombardo, H. Delingette, and N. Ayache. Improving realism of a surgery simulator: linear anisotropic elasticity, complex interactions and force extrapolation. *Journal of Visualisation and Computer Animation*, 13(3):147–167, july 2002.

[32] C. Forest, H. Delingette, and N. Ayache. Cutting simulation of manifold volumetric meshes. In *Modelling & Simulation for Computer-aided Medicine and Surgery (MS4CMS'02)*, pages 235–244, Tokyo, September 2002. Springer.

[33] C. Forest, H. Delingette, and N. Ayache. Removing tetrahedra from a manifold mesh. In *Computer Animation (CA'02)*, pages 225–229, Geneva, Switzerland, June 2002. IEEE Computer Society.

[34] G. Picinbono and J.-C. Lombardo. Extrapolation: a solution for force feedback? In *International Scientific Workshop on Virtual Reality and Prototyping*, pages 117–125, Laval France, June 3-4 1999.

[35] G. Picinbono, J.-C. Lombardo, H. Delingette, and N. Ayache. Anisotropic elasticity and forces extrapolation to improve realism of surgery simulation. In *ICRA2000: IEEE International Conference Robotics and Automation*, pages 596–602, San Francisco USA, April 2000.

[36] Herve Delingette and Nicholas Ayache. Hepatic surgery simulation. *Commun. ACM*, 48(2):31–36, 2005.

[37] D. C. Popescu and M. Compton. A model for efficient and accurate interaction with elastic objects in haptic virtual environments. In *GRAPHITE '03: Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 245–250, New York, NY, USA, 2003. ACM Press.

[38] I. Nikitin, L. Nikitina, P. Frolov, G. Goebbels, M. Göbel, S. Klimenko, and G. M. Nielson. Real-time simulation of elastic objects in virtual environments using finite element method and precomputed green's functions. In *EGVE '02: Proceedings of the workshop on Virtual environments 2002*, pages 47–52, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.

[39] G. Picinbono, H. Delingette, and N. Ayache. Non-linear anisotropic elasticity for real-time surgery simulation real-time elastic deformations of soft tissues for surgery simulation. Technical Report RR-4028, INRIA, 2000.

[40] G. Picinbono, H. Delingette, and N. Ayache. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *ICRA2001: IEEE International Conference Robotics and Automation*, Seoul Korea, May 2001. 6 pages.

[41] G. Picinbono, H. Delingette, and N. Ayache. Non-linear anisotropic elasticity for real-time surgery simulation. *Graphical Models*, 65(5):305–321, September 2003.

[42] Y. Zhuang. *Real-time simulation of physically realistic global deformations.* PhD thesis, Department of Electrical Engineering and Computer Science, UC Berkeley, 2000. Chair-John Canny.

[43] Yan Zhuang and John Canny. Real-time simulation of physically realistic global deformation. In *IEEE Vis'99 Late Breaking Hot Topics*, 1999.

[44] Y. Zhuang and J. Canny. Real-time global deformations. In *The fourth International Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 97–107. A. K. Peters, 2000.

[45] X. Wu, M. S. Downes, T. Goktekin, and F. Tendick. Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshe. In A. Chalmers and T.-M. Rhyne, editors, *EG 2001 Proceedings*, volume 20(3), pages 349–358. Blackwell Publishing, 2001.

[46] M. Hauth, J. Groß;, and W. Straßer. Interactive physically based solid dynamics. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer animation*, pages 17–27, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[47] M. Hauth and W. Straßer. Corotational simulation of deformable solids. In *Proc. WSCG 2004*, pages 137–145, 2004.

[48] M. Hauth, J. Gross, and W. Straßer. Soft tissue simulation based on measured data. In *Proc. MICCAI 2003*, page 262270, 2003.

[49] M. LeDuc, S. Payandeh, and J. Dill. Toward modeling of a suturing task. In *Graphics Interface'03 Conference*, pages 273–279, Halifax, June 2003.

[50] R. W. Webster, D. I. Zimmerman, B. J. Mohler, M. G. Melkonian, and R. S. Haluck. A prototype haptic suturing simulator. In *Proc. Medicine Meets Virtual Reality 5*, pages 567–569, Newport Beach, 2001. IOS Press.

[51] J. Lenoir, P. Meseure, L. Grisoni, and C. Chaillou. A suture model for surgical simulation. *2nd International Symposium on Medical Simulation (ISMS'04)*, pages 105–113, june 17-18 2004.

[52] L. Margetts. *Parallel Finite Element Analysis.* PhD thesis, University of Manchester, Great Britain, 2002.

[53] M. F. Adams. *Multigrid Equation Solvers for Large Scale Nonlinear Finite Element Simulations.* PhD thesis, University of California, Berkeley, 1999.

[54] C. Farhat, N. Maman, and G. Brown. Mesh partitioning for implicit computations via domain decomposition: Impact and optimization of the subdomain aspect ratio. *Int. J. Num. Meth. Eng.*, pages 38:989–1000, 1995.

[55] Y. Saad, M. Sosonkina, and J. Zhang. Domain decomposition and multi-level type techniques for general sparse linear systems. In J. Mandel, C. Farhat, and X.-C. Cai, editors, *Domain Decomposition Methods 10*, number 218, pages 174–190, Providence, RI, 1998. AMS.

[56] D.A.Keyes. Tutorial on domain decomposition methods, July 2003. Tutorial presented at 15th Internation Conference on Domain Decomposition, Berlin, http://www.mi.fu-berlin.de/dd15/tutorial.php.

[57] H. X. Lin and H. J. Sips. Parallel direct solution of large sparse systems in finite element computations. In *ICS '93: Proceedings of the 7th international conference on Supercomputing*, pages 261–270, New York, NY, USA, 1993. ACM Press.

[58] A. Gupta, G. Karypis, and V. Kumar. Highly scalable parallel algorithms for sparse matrix factorization. *IEEE Trans. Parallel Distrib. Syst.*, 8(5):502–520, 1997.

[59] M.V. Joshi, A. Gupta, G.Karypis, and V.Kumar. A high performance two dimensional scalable parallel algorithm for solving sparse triangular systems. In *HIPC '97: Proceedings of the Fourth International Conference on High-Performance Computing*, page 137, Washington, DC, USA, 1997. IEEE Computer Society.

[60] B. Irons. A frontal solution program for finite-element analysis. *Int. J. Num. Meth. in Eng. 2*, pages 5–32, 1970.

[61] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear. *ACM Trans. Math. Softw.*, 9(3):302–325, 1983.

[62] I.S. Duff and J..A. Scott. A frontal code for the solution of sparse positive-definite symmetric systems arising from finite-element applications. *ACM Trans. Math. Softw.*, 25(4):404–424, 1999.

[63] J.A. Scott. Parallel frontal solvers for large sparse linear systems. *ACM Trans. Math. Softw.*, 29(4):395–417, 2003.

[64] I.S. Duff and J.A. Scott. A parallel direct solver for large sparse highly unsymmetric linear systems. *ACM Trans. Math. Softw.*, 30(2):95–117, 2004.

[65] N. I. M. Gould and J. A. Scott Y. Hu. A numerical evaluation of sparse direct solvers for the solution of large sparse, symmetric linear systems of equations. Technical report, Council for the Central Laboratory of the Research Councils, UK, 2005.

[66] N. I. M. Gould and J. A. Scott Y. Hu. Complete results from a numerical evaluation of sparse direct solvers for the solution of large, sparse, symmetric linear systems of equations. Technical report, Council for the Central Laboratory of the Research Councils, UK, 2005.

[67] S. Jo Kim, C. Sung Lee, J. Ho Kim, M. Joh, and S. Lee. IPSAP: A high-performance parallel finite element code for large-scale structural analysis based on domain-wise multifrontal technique. In *SC '03: Proceedings of the 2003 ACM/IEEE conference on Supercomputing*, page 32, Washington, DC, USA, 2003. IEEE Computer Society.

[68] I. C. F. Ipsen and C. D. Meyer. The idea behind Krylov methods. *American Mathematical Monthly*, 105(10):889–899, 12 1998.

[69] V. Eijkhout. Overview of iterative linear system solver packages. Technical report, University of Tennessee, Knoxville, TN, USA, 1998.

[70] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 1994.

[71] Y.Saad and H.A. van der Vorst. Iterative solution of linear systems in the 20th century. *J. Comput. Appl. Math.*, 123(1-2):1–33, 2000.

[72] Shao-Liang Zhang. GPBi-CG: Generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems. *SIAM J. Sci. Comput.*, 18(2):537–551, March 1997.

[73] G.A. Gravvanis and K.M. Giannoutakis. Parallel approximate finite element inverse preconditioning on distributed systems. In *ISPDC '04: Proceedings of the Third International Symposium on Parallel and Distributed Computing/Third International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (ISPDC/HeteroPar'04)*, pages 277–283, Washington, DC, USA, 2004. IEEE Computer Society.

[74] G.Karypis and V.Kumar. Multilevel k-way partitioning scheme for irregular graphs. *J. Parallel Distrib. Comput.*, 48(1):96–129, 1998.

[75] X. Cai and Å. Ødegård. Parallel simulation of 3d nonlinear acoustic fields on a linux-cluster. In *Proceedings of the Cluster 2000 Conference*, 2000. 8 pages.

[76] R. Blaheta and P. Byczanski et. al. Large scale parallel FEM computations of far/near stress field changes in rocks. *Future Gener. Comput. Syst.*, 22(4):449–459, 2006.

[77] K. Kashiyama. Large scale finite element simulation and modeling for wind flow and rainfall. In *HPCASIA '04: Proceedings of the High Performance Computing and Grid in Asia Pacific Region, Seventh International Conference on (HPCAsia'04)*, pages 404–410, Washington, DC, USA, 2004. IEEE Computer Society.

[78] K. Pierson M. Lesoinne. FETI-DP: An efficient, scalable and unified dual-primal feti method. In *Proceedings of the 12th International Conference on Domain Decomposition Methods*, 1999.

[79] K.H. Pierson, G.M. Reese, and P. Raghavan. Experiences with FETI-DP in a production level finite element application. In *Proceedings of the 14th International Conference on Domain Decomposition Methods*, 2002.

[80] M. Bhardwaj, K. Pierson, G. Reese, T. Walsh, D. Day, K. Alvin, J. Peery, C. Farhat, and M. Lesoinne. Salinas: a scalable software for high-performance structural and solid mechanics simulations. In *Supercomputing '02: Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–19, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[81] C. Farhat and J. Li. An iterative domain decomposition method for the solution of a class of indefinite problems in computational structural dynamics. *Appl. Numer. Math.*, 54(2):150–166, 2005.

[82] J. E. Jones. Parallel multigrid tutorial, 1999. Tutorial presented at Copper Mountain Conference on Multigrid Methods, Colorado, USA.

[83] V. E. Henson. An algebraic multigrid tutorial, 1999. Tutorial presented at Copper Mountain Conference on Multigrid Methods, Colorado, USA.

[84] M. Sala, J. Hu, and R. S. Tuminaro. Ml 3.1 smoothed aggregation user's guide. Technical report, Sandia National Laboratories, 2004.

[85] P. Vanek, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. Technical Report UCD-CCM-036, Center for Computational Mathematics, University of Colorado, 1995.

[86] P. Vanek, M. Brezina, and J. Mandel. Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik*, 88(3):559–579, 2001.

[87] M. Adams and J. W. Demmel. Parallel multigrid solver for 3d unstructured finite element problems. In *Supercomputing '99: Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM)*, page 27, New York, NY, USA, 1999. ACM Press.

[88] M. F. Adams, H. H. Bayraktar, T. M. Keaveny, and P.Papadopoulos. Ultrascalable implicit finite element analyses in solid mechanics with over a half a billion degrees of freedom. In *SC '04: Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, page 34, Washington, DC, USA, 2004. IEEE Computer Society.

[89] I.M. Llorente, M.Prieto-Matas, and B.Diskin. An efficient parallel multigrid solver for 3-d convection-dominated problems. Technical report, NASA Langley Research Center, 2000.

[90] M.Prieto, R.S. Montero, and I.M. Llorente. A parallel multigrid solver for viscous flows on anisotropic structured grids. Technical report, NASA Langley Research Center, 2001.

[91] X. Zhao, P. G. Richards, and S. J. Zhang. A parallel algorithm for algebraic multigrid. In *ACM-SE 42: Proceedings of the 42nd annual Southeast regional conference*, pages 285–290, New York, NY, USA, 2004. ACM Press.

[92] V.Akcelik, G.Biros, and O.Ghattas. Parallel multiscale gauss-newton-krylov methods for inverse wave propagation. In *Supercomputing '02: Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–15, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[93] B.Butrylo, C.Vollaire, and L.Nicolas. Parallel implementation of the vector finite element and finite difference time domain methods. In *PARELEC '02: Proceedings of the International Conference on Parallel Computing in Electrical Engineering*, page 347, Washington, DC, USA, 2002. IEEE Computer Society.

[94] Shripad Thite. A unified algorithm for adaptive spacetime meshing with nonlocal cone constraints. In *21st European Workshop on Computational Geometry (EWCG)*, pages 1–4, Eindhoven, Netherlands, March 2005.

[95] Y. Zhou, M. Garland, and R. Haber. Pixel-exact rendering of spacetime finite element solutions. In *VIS '04: Proceedings of the conference on Visualization '04*, pages 425–432, Washington, DC, USA, 2004. IEEE Computer Society.

[96] R. Abedi, S. Chung, J. Erickson, Y. Fan, M. Garland, D. Guoy, R. Haber, J. Sullivan, S. Thite, and Y. Zhou. Spacetime meshing with adaptive refinement and coarsening. In *Proc. 20th Symp. Computational Geometry*, pages 300–309, June 2004.

[97] Christian Wieners. Distributed point objects. a new concept for parallel finite elements. In *Lecture Notes in Computational Science and Engineering 40*, pages 175–183, 2003.

[98] A. Křenek. *Towards Interactive Molecular Models.* PhD thesis, Faculty of Informatics, Masaryk University in Brno, Czech Republic, 2005.

[99] P. Burda B. Sousedík. Finite element formulation of three-dimensional nonlinear elasticity problem. Technical report, Seminar in Applied Mathematics, Czech Technical University Prague, 2005.

[100] J. W. Romein, A. Plaat, H. E. Bal, and J. Schaeffer. Transposition table driven work scheduling in distributed search. In *AAAI/IAAI*, pages 725–731, 1999.

List of Author's Publications and Presentations

Publications

[1] Igor Peterlík and Luděk Matyska. An Algorithm of State-Space Precomputation Allowing Non-linear Haptic Deformation Modelling Using Finite Element Method *Second Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems.* Tokyo, 2007, 6 pages, accepted

[2] Igor Peterlík, Aleš Křenek Haptically Driven Travelling Through Conformational Space. *F*irst Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. Pisa: IEEE Computer Society, 2005,. pages 342–347, ISBN 0-7695-2310-2

[3] Aleš Křenek, Igor Peterlík and Luděk Matyska. Building 3D State Spaces of Virtual Environments with a TDS-based Algorithm *R*ecent Advances in Parallel Virtual Machine and Message Passing Interface. Berlin: Springer-Verlag, 2003,. pages 529–536, ISBN 3-540-20149-1

[4] Aleš Křenek, Igor Peterlík. Distribuované výpočty složitých stavových prostorøu *Širokopásmové sítě a jejich aplikace.* Olomouc : CESNET, z.s.p.o., 2005,. pages 176-185, ISBN 80-244-1035-4

Technical Reports

[5] Aleš Křenek, Igor Peterlík. Haptically Driven Travelling Through Conformational Space *Technical Report, Faculty of Informatics, Masaryk University*, FIMU-RS-2005-02, 2005, 22 pages

Posters

[6] Igor Peterlík. Soft Tissue Modelling with State Space Precomputation *Poster Session of Informatic Seminar*, Faculty of Informatics, Masaryk University, 2006

Presentations

[7] Igor Peterlík. Haptically Driven Travelling Through Conformational Space. *F*irst Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. Pisa, 2005

[8] Igor Peterlík. Distributed Computations of Large State Spaces *MEMICS05, 1st Doctoral Workshop on Mathematical and Engineering Methods in Computer Science*, Znojmo, 2005

[9] Igor Peterlík. Finite Element Method and Soft Tissue Modelling *Informatic Seminar*, Faculty of Informatics, Masaryk University, 2005

Appendices

Appendix A

An Algorithm of State-Space Precomputation Allowing Non-linear Haptic Deformation Modelling Using Finite Element Method

by Igor Peterlík, Luděk Matyska

In *Second Joint EuroHaprics Conference and Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems.* Los Alamitos : IEEE Computer Society Press, 2007. pg. 231–236, ISBN 0-7695-2738-8.

An Algorithm of State-Space Precomputation Allowing Non-linear Haptic Deformation Modelling Using Finite Element Method*

Igor Peterlík   Luděk Matyska
Faculty of Informatics, Brno, Czech Republic
E-mail: xpeterl@fi.muni.cz, ludek@ics.muni.cz

Abstract

After presenting the mathematical background of modelling elastic deformations together with finite element formulation, we propose a new algorithm allowing haptic interaction with soft tissues having both non-linear geometric and physical properties.

The algorithm consists of two phases — first the configuration space is precomputed in a high-performance possibly distributed environment, then the precomputed data are used during the haptic interaction.

In the paper we focus mainly on sequential description of the first phase of the algorithm. We present some preliminary experimental results concerning the accuracy of the proposed method as well and the sketch of further extensions.

1. Related Work and Our Goals

Virtual interaction with deformable objects is an attractive, however computationally demanding area of research. Employing the recent technologies developed within the frame of the virtual reality such as stereo visualisation and mainly haptics make the interaction much more realistic and allows the user to "feel" the objects she is interacting with. This imposes much higher requirements on the speed and accuracy of computations when simulating the behaviour of the modelled objects. E. g. whereas the refresh rate needed for realistic visualisation is about 25 Hz, the refresh rate required in haptic rendering is usually above 1 kHz due to the much higher sensitivity of the human touch senses.

On the other hand, the models should be convincing and therefore based on real physics. The equations from the theory of elasticity are usually employed when establishing the mathematical formulation of the problem that is finally solved by some complex method such as finite elements. It is a well-known fact that performing this computations within haptic loop is far beyond the capabilities of nowadays computers.

There have been several attempts to address this issue such as simplification of the underlying models or employing some precomputations before the real interaction occurs. In [1], linearised model is used with precomputation of elementary displacement. The model is further extended in [2] by topology changes based on *mass-tensors*. Further, non-linear model is proposed decreasing the refresh rate to 25 Hz [3]. Linear model together with haptic interaction is described in [4] applying *small area paradigm*. The model with non-linear geometry is described in [5], several techniques as mass lumping are applied, however again only the visual refresh rate is achieved. In [6] both geometrical and physical non-linear models are proposed, but no results about the speed and refresh rates are given.

When using the simplified linearised models, remarkable artefacts appear and only very small deformations are rendered realistically. Therefore, we looked for new approach allowing the haptic interaction with the non-linear models. Applying the finite element method, which is commonly used in problems of this type, we arrive in large systems of non-linear algebraic equations which must be solved iteratively. Therefore, we focus on techniques based on precomputation and we followed up our work published in [7] where we described an algorithm for the state-space precomputation allowing the haptic interaction with biomolecules.

In this paper, we present an algorithm based on our previous work applied in the area of deformation modelling, especially in simulations concerning the soft tissues. First, we give the mathematical background from the theory of elasticity and formulate the problem using the finite element method. The main part is dedicated to the precomputation algorithm and some modifications of classical FE formulation when employing the haptics. So far we have partially performed the proof-of-concept implementation of the algorithm, therefore we briefly present some practical results and finally, we sketch the future extensions of the algorithm.

*Supported by Ministry of Education of the Czech Republic, research intents MSM6383917201 and MSM0021622419.

2. Mathematical Background in Deformation Modelling

In this section we briefly present the mathematical background of the deformation modelling of the soft tissues. First we introduce the underlying physical entities and sketch the formulation of the problem in the theory of elasticity. Then we employ the finite element method and finally we show two basic methods for solving large system of non-linear equations.

2.1. Physical Representation of Deformations

There are two basic physical quantities taking part in the process of deformation — *applied forces* and *displacement*. The theory of elasticity provides a mathematical tool, which relates these external entities using their internal counterparts.

The internal entity corresponding to the displacement is the *strain tensor*. There are several strain tensors used in the theory of elasticity, however our formulation is based on the non-linear Green-St.Venant strain tensor $\mathbf{E}$, which is defined as

$$\mathbf{E} = \nabla\mathbf{u} + \nabla\mathbf{u}^T + \nabla\mathbf{u}^T\nabla\mathbf{u} \tag{1}$$

The internal quantity associated with the forces is the *stress tensor*. Our formulation uses the second Piola-Kirchhoff stress tensor $\sigma(\mathbf{x})$.

The constitutive equation defines the relation between the applied forces and the displacement by coupling the strain and the stress tensors. Moreover, it determines the physical properties of the material.

We employ the hyper-elastic type of the material based on *stored energy function* $\mathbf{W}(\mathbf{E})$ which couples the strain and stress tensor by:

$$\sigma(\mathbf{E}) = \frac{\partial\mathbf{W}(\mathbf{E})}{\partial\mathbf{E}} \tag{2}$$

The particular formulation of the function W depends on the chosen material law. So far, we have employed St.Venant material. However, we plan to employ some other widely used materials, such as Mooney-Rivlin.

The full mathelatical formulation of the deformation problem gives a governing partial differential equation which can be non-linear due to the non-linearity of the strain tensor of the constitutive equation. The further non-linearity can be brought in by using some more complex material law.

2.2. Finite Element Formulation

To solve the problems emerging in the theory of elasticity, the *finite element method* is usually applied. The method reduces the complex partial differential equation into a large system of algebraic equations which can be solved by some numerical method.

The strategy of the method is to discretize the continuous domain of the deformed body by a mesh of elements with a simple geometry and compute the approximate solution in the nodes of the mesh. The advantage of the method is in providing the interpolation functions that can be used for approximating the solution over the volume of the elements. [8]

Since we use the non-linear strain tensor 1, the finite element formulation gives us the non-linear system

$$\mathbf{A}(\mathbf{u}) = \mathbf{f} \tag{3}$$

where $\mathbf{A}(\mathbf{u})$ is the stiffness matrix defined as

$$\mathbf{A}(\mathbf{u}) = \int_{\Omega^e} \frac{\partial W}{\partial E_{jk}} \left(\delta_{ij} + \frac{\partial u_i}{\partial x_j}\right) \frac{\partial w_i}{\partial x_k} dx \tag{4}$$

and $\mathbf{f}$ is the load vector

$$\mathbf{f} = \int_{\Omega^e} fw dx + \int_{\partial\Omega^e} gw da. \tag{5}$$

2.3. The Numerical Solution

The above system can be solved using an iterative method. Generally, in each iteration, having the solution estimation $\mathbf{u}^{(n)}$, the new estimation is computed as

$$\mathbf{u}^{(n+1)} = \mathbf{u}^{(n)} + \delta\mathbf{u}^{(n+1)} \tag{6}$$

The displacement increment $\delta\mathbf{u}^{(n)}$ is computed from linearised system depending on the method being used. In *method of incremental loads*, we let the body force vary by a small force increment

$$\delta\mathbf{f}^{(n)} = (\lambda^{(n+1)} - \lambda^{(n)})\mathbf{f}, \qquad 0 \geq \lambda^{(n)} \geq 1 \tag{7}$$

and then, we compute the displacement increment from linear system

$$\mathbf{A}'(\mathbf{u}^{(n)})\delta\mathbf{u}^{(n+1)} = \delta\mathbf{f}^{(n)} \tag{8}$$

where $\mathbf{A}'(\mathbf{u})$ is *Fréchet* derivative or tangent stiffness matrix that is defined as derivative of the stiffness matrix with respect to the variables (for the exact formulation see [10]).

In *Newton* method, the linearised system is of the form

$$\mathbf{A}'(\mathbf{u}^{(n)})\delta\mathbf{u}^{(n)} = \mathbf{f} - \mathbf{A}(\mathbf{u}^{(n)}) \tag{9}$$

where $\mathbf{A}(\mathbf{u}^{(n)})$ is defined above.

Both methods can be used simultaneously — the method of incremental loads is used as a predictor and the Newton method as the corrector as follows: we start the computation with zero force vector and we increase the force in successive steps computing the displacement by the incremental load method. After several steps (depending on the size of residual), we perform correction step by using Newton method taking the actual displacement vector as the initial estimation. This process is iteratively repeated until the desired force vector is achieved and corresponding displacement computed with requested residual.

3. Haptic Model With State Space Precomputation

3.1. The Displacement-Driven Interaction and Lagrange Multipliers

As introduced in section 2.1, the standard deformation modelling helps us to formulate and solve the problem of computing a deformation (more accurately the displacement) of a body after the forces have been applied. However, the situation in haptics is slightly different. Controlling a haptic device we interact with the soft body by touching its surface by some probe (a virtual representation of the haptic interaction point) displacing a part of the surface and we need to compute the reacting force and the entire deformation. This approach is call *displacement driven interaction.*

Further we expect the FEM discretization to be already computed and by interacting with the virtual body we mean the interaction with its final element mesh. Then the interaction works as follows:

- we move the virtual probe towards the modelled body
- for the actual position of the probe, we perform collision detection between the probe and the FE mesh, if there is some, we get the displacement of the nodes *directly touched* by the probe
- having the set of the touched displaced nodes, we compute the deformation of the entire body as well as the reaction force acting on the probe

To cope with this modification of the problem we apply *Langrange multipliers* approach on the system of non-linear equations $\mathbf{A}(\mathbf{u}) = \mathbf{f}$ derived is section 2.2. For the sake of simplicity we suppose there is only the i-th node touched by the probe and its displacement is $u_i = t$ known as *pseudo-load.* The force f_i exerted in this point is unknown and we denote it by a variable h. This means the i-th equation of the non-linear system is modified as:

$$\mathbf{A}_i(u_1, \dots u_n) = h \tag{10}$$

After simple operation we have a modified equation having $n + 1$ variables:

$$\hat{\mathbf{A}}_i(u_1, \dots u_n, h) = \mathbf{A}_i(u_1, \dots u_n) - h = 0 \tag{11}$$

Since we have added a new variable, we have to add a new equation to the original system $\mathbf{A}$. This is simple, since we know the prescribed displacement t for the i-th node. Therefore the new equation is:

$$\hat{\mathbf{A}}_{n+1}(u_1, \dots u_n, h) = u_i = t \tag{12}$$

If there are prescribed displacements (the pseudo-loads) of $k > 1$ nodes, the technique is analogical:

- the vector of variables $\mathbf{u}$ is augmented by the unknown forces $\mathbf{h}$ (denoting $\mathbf{u}|\mathbf{h}$)
- the right hand-side vector of known forces $\mathbf{f}$ is augmented by the pseudo-loads $\mathbf{t}$
- the equations corresponding to the touched nodes are modified analogically as described above
- k new equations are added to the original system A putting the displacements of the touched nodes to the prescribed values

After applying the modifications we get the system

$$\hat{\mathbf{A}}(\mathbf{u}|\mathbf{h}) = \mathbf{f}|\mathbf{t}. \tag{13}$$

3.2. The Precomputation Algorithm and Interaction

3.2.1. Motivation for the Precomputation Algorithm

To achieve the desired accuracy of the finite element modelling of the deformable body, the discretization to the finite elements must be fine enough. This makes the number of variables to be really large as well as the number of the equations in the system $\mathbf{A}(\mathbf{x}) = \mathbf{f}$. In practice, the number can vary from thousands to millions. Therefore even in case of the linearised problems, it is difficult to keep pace with the speed of the haptic loop (where the frequency in order of kHz is requested). In case of the non-linear problems, which we are interested in, the computation of the deformations is not feasible, since the numerical methods for solving such large systems of non-linear equations are iterative.

The interaction between the probe and the soft body can be regarded as travelling through continuous infinite-state configuration space, where the configuration is given by the position of the probe, the displacement of the touched and free nodes and the reacting applied force. Since the on-line computations are not feasible, we designed an algorithm that precomputes a discretized finite configuration space and applies the precomputed data in the interaction approximating any desired configuration using interpolation.

3.2.2. The Idea of the Algorithm

More precisely, having the deformation and applied force precomputed for some given position of the probe, we can compute an approximation of the displacement and force in the vicinity of this precomputed point. Therefore, we can precompute some predefined set of configurations and use interpolation methods to compute any intermediate configuration in sufficiently short time.

Therefore we introduce a new discretization of the space, where the soft body is placed. To distinguish this discretization from the one used in finite element method (see section 2.2), we call this *space discretization.* It is performed by creating tree-dimensional regular grid covering the entire space where the soft body is placed. The

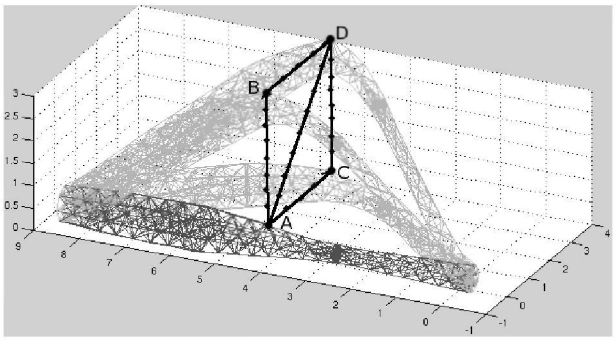

Figure 1: Four configurations of the system with deformable body — starting in the configuration A, a point on the surface of the body was displaced to positions B,C and D

grid consists of nodal points called *check-points* and paths between the adjacent check-points.

The first idea is to put the probe into each check-point of the 3-D grid and to compute the configuration for this position. However, this is not sufficient, as the actual configuration depends not only on the actual position of the probe, but on the history of the travelling as well, e. g. if the soft body is represented by a vertical plate perpendicular to the x axis, the deformation is different when hitting the plate by the probe from the left and from the right. Therefore, instead of instant placing the probe into a check-point, we compute the transitions between adjacent check-points, starting in a check-point outside the soft body. In each check-point we establish *levels of configurations* and each time we traverse the path from the source check-point to the target, we compare the actual configuration with those already stored in the target and if it differs from each of them by some given factor, we add a new level and store the configuration there. In addition, we also store the number of the level reached in the target check-point to a data structure in the source check-point.

As the number of transitions in the finite grid is bounded and the number of the levels in each check-point is restricted as described above, the precomputed space is finite and the computations eventually terminate.

Not all the configurations of the body are reachable in practice, since the feedback force applied to the haptic device is restricted to some interval due to mechanical construction of the device. This fact can be used to establish some bound on the precomputation of the space as follows: if the value of the force of a new computed configuration lies inside the restricted interval, it is stored in a new level which is marked for further expansion. This means the configuration will be later used as a source state for computing the transitions to the adjacent check-points. Otherwise, the configuration is stored as well, however it is not expanded anymore.

In the next section, we describe the process of traversing the configuration space in more detail. We consider the interaction to be non-destructive, i. e. the topology of the soft body does not change.

3.2.3. The Precomputation Phase in Detail

In this section, we mathematically describe the travelling through the configuration space during the precomputation phase.

The space discretization grid is defined as a set of check-points $\{P_{ijk} | 1 \leq i \leq N_x, 1 \leq j \leq N_y, 1 \leq k \leq N_z\}$, where the N_x, N_y and N_z are the dimensions of the grid. The configuration C_{ijk} stored in level l at the check-point P_{ijk} is defined as a tuple $(\mathbf{t}_C, \mathbf{u}_C, \mathbf{h}_C, \mathbf{l}_C, toExp_C)$, where $\mathbf{t}_C$ is the displacement vector of the touched nodes computed by a collision detection, $\mathbf{u}_C$ is the displacement vector of the free nodes and $\mathbf{h}_C$ is the reaction force. The vector $\mathbf{l}_C$ and the flag $toExp_C$ does not concern with the physical scenario of the interaction (the position of the probe, deformation etc.). If the state is reachable (the force vector is acceptable by the haptic device), then $toExp = 1$ and the configuration will be expanded in the future, otherwise $toExp = 0$. If the configuration is marked for further expansion, the vector $\mathbf{l}$ stores the numbers of levels which are achieved when the transitions to the adjacent points are computed. Initially, the vector is set to some undefined value and is updated during the expansion of the configuration.

The transition between the check-points introduced in previous section is defined as follows: having an already computed source configuration C_{ijk} for the probe position P_{ijk} stored in level l, compute the new target configuration $C'_{\overline{ijk}}$ for a position of probe in point P'_{ijk} such that $||ijk - \overline{ijk}|| \in \{1, \sqrt{2}, \sqrt{3}\}$ and store it in some new level l'.

For the sake of simplicity, we denote A the source check-point having a configuration $C_A = (\mathbf{t}_A, \mathbf{u}_A, \mathbf{h}_A, \mathbf{l}_C, 1)$ stored in some level and B the target check-point. In the following, we describe the transition from source configuration C_A to the target B resulting in configuration C_B.

First, the path AB connecting the corresponding check-points is divided by $m+1$ points $(\lambda_{AB}^{(0)}, \dots \lambda_{AB}^{(m)})$ into m intervals, where $\lambda_{AB}^{(0)} = A$ and $\lambda_{AB}^{(m)} = B$. We traverse the path by pushing the probe from point $\lambda_{AB}^{(i-1)}$ to point $\lambda_{AB}^{(i)}$ performing the method of incremental loads, i. e. we compute the force $\mathbf{h}_{AB}^{(i)}$ and displacement $\mathbf{u}_{AB}^{(i)}$ in the i-th point path (for $i > 1$) from the actual pseudo-loads $\mathbf{t}_{AB}^{(i)}$ and results $\mathbf{h}_{AB}^{(i-1)}$ and $\mathbf{u}_{AB}^{(i-1)}$ computed in the previous point.

Initially, we put $\mathbf{u}_{AB}^{(0)} = \mathbf{u}_A$ and $\mathbf{h}_{AB}^{(0)} = \mathbf{h}_A$ and in each next point $\lambda_{AB}^{(i)}$:

- collision detection between the probe and soft body is performed resulting in $\mathbf{t}_{AB}^{(i)}$

- the tangent stiffness matrix $\mathbf{A}'(\mathbf{u}_{AB}^{(i-1)}|\mathbf{h}_{AB}^{(i-1)})$ is assembled from the results in the previous step (see section 2.3 for details)
- the increments of the displacement $\delta\mathbf{u}_{AB}^{(i)}$ and reacting force $\delta\mathbf{h}_{AB}^{(i)}$ are computed solving the linear system

$$\mathbf{A}'(\mathbf{u}_{AB}^{(i-1)}|\mathbf{h}_{AB}^{(i-1)})(\delta\mathbf{u}_{AB}^{(i)}|\delta\mathbf{h}_{AB}^{(i)}) = \mathbf{t}_{AB}^{(i)} - \mathbf{t}_{AB}^{(i-1)} \tag{14}$$

where the notation $(\mathbf{u}|\mathbf{h})$ is defined is section 3.1
- the actual displacement and reacting force are incremented

$$\mathbf{u}_{AB}^{(i)} = \delta\mathbf{u}_{AB}^{(i)} + \mathbf{u}_{AB}^{(i-1)} \tag{15}$$

$$\mathbf{h}_{AB}^{(i)} = \delta\mathbf{h}_{AB}^{(i)} + \mathbf{h}_{AB}^{(i-1)} \tag{16}$$

After computing the m loops of the incremental load method, we arrive at the target check-point $\lambda_{AB}^{(m)} = B$ having the approximations $\mathbf{u}_{AB}^{(m)}$ and $\mathbf{h}_{AB}^{(m)}$. In this moment, the correction process is performed using the Newton method. The number of Newton iteration is given by the desired accuracy of the method and cannot be determined in advance.

Denoting $\bar{\mathbf{u}}_B^{(j)}$ and $\bar{\mathbf{h}}_B^{(j)}$ the estimations of the displacement and reacting force in the j-th iteration of the Newton method, we put initially $\bar{\mathbf{u}}_B^{(0)} = \mathbf{u}_{AB}^{(m)}$, $\bar{\mathbf{h}}_B^{(0)} = \mathbf{h}_{AB}^{(m)}$ and $\mathbf{t}_B = \mathbf{t}_B^{(m)}$. Then in each following iteration:

- the tangent stiffness matrix $\mathbf{A}'(\bar{\mathbf{u}}_B^{(j-1)}|\bar{\mathbf{h}}_B^{(j-1)})$ and the stiffness vector $\mathbf{A}(\bar{\mathbf{u}}_B^{(j-1)}|\bar{\mathbf{h}}_B^{(j-1)})$ are assembled from the results computed in the previous step (see sections 2.2 and 2.3 for details)
- the corrections of the displacement $\delta\bar{\mathbf{u}}_B^{(j)}$ and reacting force $\delta\bar{\mathbf{h}}_B^{(j)}$ are computed solving the linear system

$$\mathbf{A}'(\bar{\mathbf{u}}_B^{(j-1)}|\bar{\mathbf{h}}_B^{(j-1)})(\delta\bar{\mathbf{u}}_B^{(j)}|\delta\bar{\mathbf{h}}_B^{(j)}) = \mathbf{t}_B^{(j)} - \mathbf{A}(\bar{\mathbf{u}}_B^{(j-1)}|\bar{\mathbf{h}}_B^{(j-1)}) \tag{17}$$

- the corrections are applied to the actual estimation of the displacement and the force:

$$\bar{\mathbf{u}}_B^{(j)} = \delta\bar{\mathbf{u}}_B^{(j)} + \bar{\mathbf{u}}_B^{(j-1)} \tag{18}$$

$$\bar{\mathbf{h}}_B^{(j)} = \delta\bar{\mathbf{h}}_B^{(j)} + \bar{\mathbf{h}}_B^{(j-1)} \tag{19}$$

- the residual $r^{(j)} = ||\mathbf{t}_B^{(j)} - \mathbf{A}(\bar{\mathbf{u}}_B^{(j)}|\bar{\mathbf{h}}_B^{(j)})||$ is computed

If in the k-th iteration the residual is smaller than some defined constant, i. e. $r^{(k)} \leq \epsilon$, then the Newton method terminates and the actual values of the displacement and force establish the new configuration $C_B = (\mathbf{t}_B, \mathbf{u}_B, \mathbf{h}_B)$ with $u_B = \bar{\mathbf{u}}_B^{(k)}$ and $h_B = \bar{\mathbf{h}}_B^{(k)}$.

Afterwards, the new configuration C_B is compared with all configurations D_B stored in the levels of the check point B. This comparison is performed using the residuals of differences of the vectors $\mathbf{t}$, $\mathbf{u}$ and $\mathbf{h}$. If there is some configuration stored at some level k that is sufficiently close to the new one, the vector of target levels in configuration C_A is updated with $\mathbf{l}_A(B) = k$ and the new configuration is abandoned. Otherwise, a new level j is established where $j - 1$ is the number of levels established so far, the new configuration C_B is stored there and the vector of target levels in C_A is updated with $\mathbf{l}_A(B) = j$.

Finally, the size of the force h_B stored in C_B is compared to the interval of the forces acceptable by the haptic device. If the value of the force is inside the interval, the flag $toExp_C$ is set and the elements of the vector $\mathbf{l}_C$ are set to -1. Otherwise, $toExp_C = 0$.

All around, the algorithm starts with expanding the zero configurations — those which lies in check-points that are initially outside the soft body. On the other hand, it terminates, when all configurations which have the flag $toExp$ set to 1 poses vector $\mathbf{l}$ with all elements not equal to -1.

3.2.4. The Interpolation Phase

During the interaction, the deformation and reaction force for any position of the probe are computed by interpolation of the precomputed values. For the actual position of the probe, the set of the closest configurations is chosen. The motion of the probe in space can be viewed as following a path, which is interpolated by the discrete virtual paths computed and stored during the precomputation phase. The density of the check-points with precomputed configurations determined the accuracy of the interpolations. The more precise relation between these two entities will be examined experimentally.

The interaction consists of two basic parts — the user haptically manipulates the virtual probe feeling the reacting force and the entire soft body is visualised on a display or stereo projection. Therefore, the interpolation consists of two threads — the first operating the haptic device and the second performing the visualisation.

The haptic thread runs in real-time mode with a high refresh rate. It computes the interpolation that must be smooth and precise enough so that the haptic part of the model behaves realistically. As we cope with the interpolation of three numbers (the force vector), we can afford to employ the interpolation functions of high orders getting better results still within the haptic loop.

The visualisation of the deformation is more computationally expensive, since all the degrees of freedom of the soft body must be interpolated. However, the lower refresh rate of the visualization gives us enough time to do this. Also, the accuracy of the visual perception is not so high as needed for the haptics and we can use the interpolation function of lower order.

4. Implementation and Results

We have performed a proof-of-concept implementation of the algorithm in Matlab programming language based on

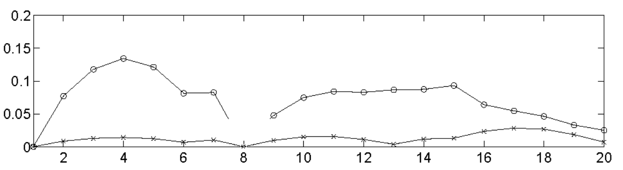

Figure 2: The relative force and deformation error of the interpolated method against the precise Newton computations of 20 steps of a random motion of the probe

the model with non-linear geometry 2.1 and material law given in 2.1. We employed some simplifications — instead of moving the probe and computing the collision detection, we grabbed some point of the finite element mesh and pulled it away from its initial position. Displacing the grabbed node to particular check-points, we performed the precomputation of the part of the configuration space following the description given in 3.2.3.

For the purpose of comparison, we performed an exact computation of a random path $\mathcal{P}$ through the undiscretized configuration space. We selected a point on the surface of the soft body and pulled it along the path $\mathcal{P}$ in small steps. In each step, we computed the exact deformation and force by the Newton method.

Then having precomputed all the check-points surrounding the path $\mathcal{P}$ and traversions among them, we followed the same path $\mathcal{P}$ by computing the force and deformation vector in exactly the same points as the precise method using linear interpolation of the precomputed values. The relative error between the interpolation-based and the precise method is depicted in figure 4.

We observed that albeit simple linear interpolation has been employed, the interpolated values are close enough to those computed by the precise method along the entire path. Taking the maximal force computed by the precise method as the base, the differences between the values computed by the two methods did not exceed 4% and the relative deformation error computed as the normed difference between the displacement vectors remains under 14%. This confirms the validity of the proposed algorithm.

5. Conclusions and Future Work

In the paper, we proposed the algorithm based on state space precomputation that allow haptic interaction with soft bodies having non-linear geometric and physical properties. The main idea of the precomputation scheme and detailed description of the computational part of the algorithm are presented.

We presented the main idea of the algorithm as well as the detailed description of the computations and some preliminary results.

In future we plan to implement the full sequential version of the algorithm together with the haptic interaction. Since the precomputation part is computationally expensive, we will implement the parallel version of the algorithm based on extended Transposition-Driven Scheduling [9]. Further, we are interested in some extensions of the algorithm incorporating the topological changes caused by cutting and tearing soft body.

References

[1] S. Cotin, H. Delingette, and N. Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions On Visual. and Computer Graphics*, 5(1):62–73, 1999.

[2] G. Picinbono, J-C. Lombardo, H. Delingette, and N. Ayache. Improving realism of a surgery simulator: linear anisotropic elasticity, complex interactions and force extrapolation. *Journal of Visual. and Computer Animation*, 13(3):147–167, 2002.

[3] G. Picinbono, H. Delingette, and N. Ayache. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *ICRA2001: IEEE International Conference Robotics and Automation*, Seoul Korea, May 2001.

[4] Dan C. Popescu and Michael Compton. A model for efficient and accurate interaction with elastic objects in haptic virtual environments. In *GRAPHITE '03: Proceedings of the 1st internat. conf. on Computer graphics and interactive techniques*, pages 245–250,NY USA, 2003. ACM Press.

[5] Yan Zhuang and John Canny. Real-time simulation of physically realistic global deformation. In *IEEE Vis'99*, 1999.

[6] Xunlei Wu, Michael S. Downes, Tolga Goktekin, and Frank Tendick. Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes. In *EG 2001 Proceedings*, vol. 20(3), pages 349–358. Blackwell Publ., 2001.

[7] Igor Peterlík, Aleš Křenek. Haptically Driven Travelling Through Conformational Space. In 1st Joint Eurohap. Conf. and Symposium on Haptic Interfaces. Pisa: IEEE Computer Society, 2005, pages 342-347.

[8] J.N.Reddy. *An Introduction to the Finite Element Method.* McGraw-Hill, 1993.

[9] Křenek, Aleš, Peterlík, Igor, Matyska Luděk. Building 3D State Spaces of Virtual Environments with a TDS-based Algorithm. In Recent Advances in Parallel Virtual Machine and Message Passing Interface. Berlin : Springer-Verlag, 2003, ISBN 3-540-20149-1. pages 529-536

[10] B. Sousedík, P. Burda. Finite Element Formulation of Three-dimensional Nonlinear Elasticity Problem Seminar in Applied Mathematics, Czech Technical University Prague, 2005

Appendix B

Distributed Precomputation of State-space for Haptic Interaction with Non-linear Model of Liver

by Igor Peterlík

In MEMICS07 *Third Doctoral Workshop on Mathematical and Engineering Methods in Computer Science.* Brno, 2007. pg. 169–176, ISBN 978-80-7355-077-6

Distributed Precomputation of State-space for Haptic Interaction with Non-linear Model of Liver

Igor Peterlík

Faculty of Informatics, Masaryk University
Botanická 68a, Brno
Czech Republic
peterlik@ics.muni.cz

Abstract. The realistic modelling of the nonlinear behavior of the soft tissues is of a paramount interest in the area of medical simulations. However, employing the haptic interaction makes any expensive real-time computation infeasible due to the high refresh rate (over 1 kHz). Therefore, we focus on a precomputation scheme when the state-space of the interaction is computed in advance and during the interaction only a simple interpolation of the precomputed data is calculated. In this paper, details of the distributed implementation of the precomputation phase are presented together with the experimental results using a non-linear model of a liver. Moreover, some further extensions are suggested in order to improve the haptic interaction as well as to shorten the length of the precomputation phase.

1 Motivation and Related Work

Interaction with deformable objects is an attractive but computationally demanding area of research. Employing the recent technology known as haptics rendering allows the users to "touch" the objects they are interacting with.

However, this imposes much higher requirements on the speed and accuracy of computations when simulating the behavior of the modelled object s. Whereas the refresh rate needed for realistic visualization is about 25 Hz, the refresh rate required in haptic rendering is usually above 1 kHz due to the much higher sensitivity of the human perception by haptics.

On the other hand, the models should be convincing and therefore based on physical reality. The relations from the theory of elasticity are usually employed when establishing the mathematical formulation of the problem that is finally solved by some complex method such as finite elements. It is a well-known fact that performing this computations in real-time, i. e. within haptic loop is far beyond the capabilities of nowadays computers.

There have been several attempts to address this issue such as simplification of the underlying models or employing some precomputations before the real interaction occurs. In [3], linearized model is used with precomputation of elementary displacement. Further, non-linear model is proposed decreasing the refresh rate to 25 Hz [4]. Linear model together with haptic interaction is described in [5] applying *small area paradigm*. The model with non-linear geometry is described in [6], several techniques as mass lumping are applied, however again only the visual refresh rate is achieved. In [7] both geometrically and physically non-linear model is proposed, but no results about the speed and refresh rates are given. Recently, in [8] a new approach based on precomputation is presented. During the precomputation coefficients of polynomials derived for St.Venant-Kirchhoff material law are calculated. Although this approach allow interaction with complex body having non-linear geometry properties, it is applicable just for one material law which is moreover linear.

The goal of our research was to propose a precomputation scheme which is capable of handle any static non-linear model providing realistic behavior. While the precomputation phase can be expensive as it can be performed in high-performance distributed environment, we require the respective haptic rendering using the precomputed data to be simple enough to be done on common desktop easily within the haptic loop.

In [1] we proposed precomputation scheme based on state-space precomputation. In this paper, we present implementation details together with the first practical experience with this technique.

2 Haptic Deformation Modelling

In this section we provide brief and informal introduction to the deformation modelling. We present the basics of the mathematical formulation and short description of the solution methods. Then we combine this theory with haptic rendering.

2.1 Mathematical Model and Solution Method

In the following text we present a brief description of the mathematical model together with the numerical solution method we are using in our approach. As the realistic behavior of the model is the main goal of the design, the model is fully based on the theory of the elasticity. To establish the relations, internal quantities are introduced — *strain tensor* quantifying the internal strain caused by the displacement and *stress tensor* describing the stress inside the body arising after the external forces are applied. In our model we use non-linear Green strain tensor allowing large deformations.

These two quantities are coupled by *material law* which in fact determines the property of the material (rubber, tissue...). We use two material laws — linear StVenant-Kirchhoff and non-linear Mooney-Rivlin with two material constants.

The mathematical formulation of the model results in partial differential equations (PDE) which are then supplied by boundary conditions. These non-linear equations are usually not solvable analytically even for simple cases (e. g. for regular geometries) and there are various numerical methods used for construction of an approximated solution.

Perhaps the most known and widely used method is the *finite element method* (FEM) which discretizes the body by a 3D mesh. Withing the frame of this method, the problem is reformulated resulting in large system of algebraic equations. The solution then consists of two phases — first the system must be assembled and then the solution must be found.

In the case when the original PDEs are non-linear the emerging algebraic equations are non-linear as well. The techniques for solving such a systems are usually iterative, based on Newton methods when in each step the full linearized system is assembled using the estimation of the solution from the previous iteration and a new correction is computed by solving this linearized system. However, in this case the convergence strongly depends on the initial estimation. Therefore, usually *incremental loading* approach is applied — starting from rest state with no deformation the force is increased in sufficiently small steps and in each step complete Newton method is iteratively computed. As result one gets the resulting deformation for the applied force as well as the states lying on the path from the initial to the target state.

2.2 Haptic Interaction

The usual setting in the deformation modelling is following: given the external forces applied on the surface, compute the deformation of the body fixed in space by the boundary conditions. However, in haptics the situation is different: the haptics device displaces some part of the body surface and we need to compute the overall deformation of the body and reacting forces. This concept is known as *displacement driven interaction* and is usually implemented by Lagrange multipliers. Details about employing this technique can be found in [1].

Further in this article we consider *point interaction* — using the haptic device, we can deform the soft object by attaching to one of the nodes on the surface of the FEM mesh and displacing it from its rest position. In following text we call this node *active*. In this article we assume the every time there is at most one active node.

Putting it together, the input of the algorithm is the prescribed displacement given by the position of the haptic device. The data to be computed are the reacting force (needed in haptic refresh rate) and the overall deformation (need in visual refresh rate).

3 Precomputation Scheme

As stated in the motivation the refresh rate of the haptic interaction is high. Therefore, the expensive computations described in the previous section cannot be done within the haptic loop. We proposed a precomputation scheme which provides a solution to this problem. Bellow, we present the main idea of the precomputation scheme based on state-space construction together with more detailed description of the computations.

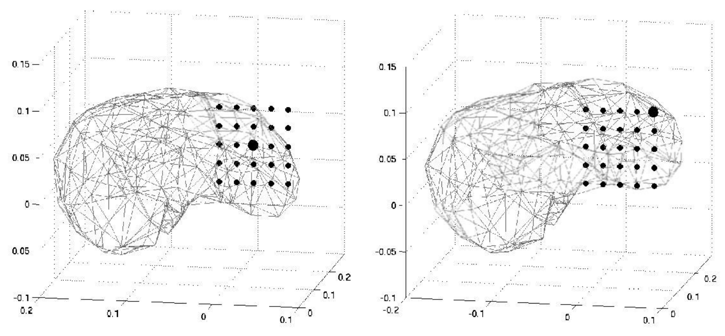

Fig. 1. Two configurations of the liver model. On the left, the active node (depicted by large black bullet) is in its rest position. On the right, the active node is displaced to one of the 25 grid points (a two-dimensional fraction of the state space represented by smaller bullets).

3.1 State-space Discretization

As the first step towards the precomputation scheme, we define a configuration of the interaction. It is given by

- the actual value of the prescribed displacement of the active node,
- the value of the reacting force acting in this node
- displacement of the other nodes in finite element mesh of the organ, i. e. the overall deformation

Then, for the particular active node we introduce a configuration space (or state-space) which contains the configurations of the model for any position of the active node. Now as we are displacing the active node from its rest position during the interaction, we are just travelling through this configuration space.

Apparently, this configuration space is infinite and continuous as there are infinitely many possible displacements of the active node. To be able to do the precomputation we proposed a construction of a subspace which fulfills two basic requirements:

- the subspace is finite and can be precomputed in finite time
- the original continuous space can be approximated using the precomputed subspace

The finite subspace is constructed by discretization of the original continuous configuration space by three-dimensional regular grid. Only the configurations in the points of this grid are computed and during the haptic interaction, the configuration for arbitrary displacement of the active node is interpolated from the precomputed data. In the picture 1, there are depicted two configurations of the liver model together with a two-dimensional fraction of the regular state-space grid.

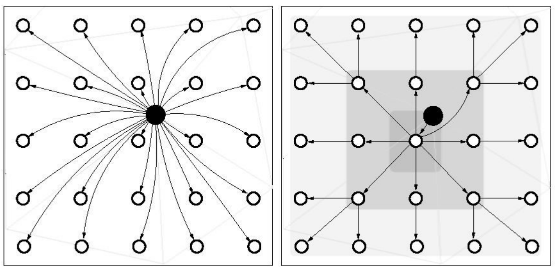

Fig. 2. Comparison of the naive and layer approach for the construction of the state-space. The black point represents the active node in rest positions, the circles denote the grid point where the configuration are to be computed. In both cases, the paths to be computed are depicted by arrows and on the right picture, the layers are shaded.

3.2 Precomputation in Details

In this subsection we focus on the construction of the state space for an arbitrary but fixed active node. Before describing the approaches to the construction of the state space, let us mention three assumptions.

First, not all the configurations are needed as some of them are not accessible during the haptic interaction. This is caused by the fact that each haptic device has limited range of forces which can be delivered to it. Therefore, too large deformations resulting in forces which exceeds the limitations of the device can be excluded from the precomputed space. This is done by introducing a bounding box around the active node and only the configurations corresponding to the displacements of the active node within the bounding box are computed. The size of the bounding box is set manually in advance and it is required that it is possible to increase its size by computing a set of new configurations and using those already computed.

Second, for further extensions in future we do not put any restrictions on the placement of the regular grid. Therefore, the rest position of the active node does not have to coincide with any point of the grid.

Third, we do not consider any material properties such as plasticity or viscoelasticity and also in this model we do not allow any topological changes. Therefore, the state space corresponding to the active node is flat, i. e. for any displacement of the active node there is only one corresponding configuration.

The naive approach to the precomputation of the configuration space corresponding to the active node is depicted in the left part of the figure 2. For each grid point within the bounding box, we displace the active node along the path going from the rest position to the target displacement associated with the grid point. It holds that the further the grid point from the rest position is, the longer the computation will take as it has to pass through more intermediate states due to the convergence.

A more efficient way how to precompute the state space within the bounding box is based on the fact that the state space is flat. The grid points are organized in *layers* as depicted on the right part of the figure 2. The computations is done as follows:

- establish the first layer of the grid by finding a grid point such that its distance from the rest position of the active node is minimal. Compute the corresponding configuration by displacing the active node to this grid point.
- the grid points adjacent to the point in the first level constitute the second layer. Starting in the first-layer configuration, compute the configurations corresponding to the points in the second layer (displace the active node to each of them).
- recursively, construct a new layer from the grid points adjacent to the points in layer constructed in the previous step and compute the new configurations using those in the previous layer.

In this case there is the same number of paths as in the previous approach, however all the paths has the same length (except the first one which is shorter) and therefore the overall computation is much faster. Moreover, the enlargement of the bounding box is not so computationally expensive, since it just adds one more layer and the new paths are starting in the outermost computed configurations.

4 Implementation, Computations and Results

4.1 Distributed Precomputation of State Spaces

We implemented the precomputation scheme using the layer approach. The configurations are computed using *Getfem* library which provides FEM code for the deformation modelling. We use both Mooney-Rivlin and StVenant-Kirchhoff material laws. The system is solved by a simple solver based on Newton method and each path is divided in four steps by incremental loading.

Since our goal was to be able to attach to any node on some part of the surface during the configuration, we implemented environment where the state spaces corresponding to the selected active nodes are computed. As the computations of distinct state spaces are independent, they are computed in distributed manner. Since each configuration is saved immediately after it is computed and can be used later for the computation another path, the construction of the state-space can be held anytime or migrated to another computation resource. Also a new layer can be added to the existing configurations.

4.2 Haptic Interaction with the Model

The haptic interaction consist of two phases. First, the user moves the free haptic interaction point (HIP) towards the model of liver. After a collision is detected, the closest node of the FEM mesh is found and HIP is attached to it. As the collision detection algorithm is beyond the scope of the paper, we do not go to the details here.

After the HIP is attached to the node, the node becomes active and corresponding precomputed state-space is used. Now, the configuration (i. e. the

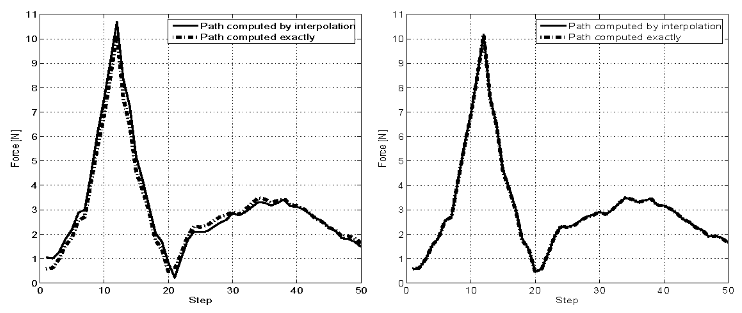

Fig. 3. Comparison of linear (left) and cubic (right) interpolation.

reacting force and overall deformation) for any displacement of the active node is computed by interpolation of the precomputed data. The computation of the interpolation runs in two threads — slower visualization thread (25 Hz) which computes the interpolation of the displacement for all the visible surface nodes and faster haptic thread (1 kHz) which computes only the interpolation of small force vectors having three elements.

We applied simple trilinear interpolation. For the actual position of the HIP, we select the closest grid points and interpolate the corresponding configurations. For verification purposes, we implemented cubic interpolation using Matlab.

4.3 The Computations and Results

For the computations, we used a three-dimensional model of pig liver [2]. The mesh was generated by *Tetgen* algorithm resulting in 281 nodes and 983 elements. Due to the convergence issues in Getfem, the second order mesh and FEM were used resulting in system with 5400 degrees of freedom.

The configurations spaces for 29 surface nodes were constructed, each construction was run on one processor. The computation of one state space took approximately 80 hours on processor Xeon 2.4 GHz. This means that the entire computations took about 2400 hours of the machine time. The resulting state space stored in simple text mode took about 180 MB of disc space and could be easily loaded into the memory during the interpolation phase.

For validation purposes, we generated several dozens random paths through the configuration space. For each path, we computed the displacements along the path precisely in small step using the Newton method in each step. Computation of each path took from 2 to 5 hours. Then we computed the configurations along the paths using the interpolation of the precomputed data. In the picture 3, there are results for two particular paths using simple linear and more advanced cubic interpolation. For comparison we compute relative error as the ration of the difference between the interpolations and the precise results. Generally, the relative error of the interpolation with respect to the precise computations were under 12% when using linear interpolation whereas employing the cubic one, the error dropped under 2%.

5 Conclusion and Future Work

In this paper we presented the precomputation based approach to haptic modelling of complex deformable objects together with some details of the implementation. As the core of the computations is the solution of the large non-linear system with more than 5000 degrees of freedom, the construction of the state spaces would be almost infeasible when not running in distributed environment.

In the future, we would like to first focus on the optimization of the solver as we believe that the convergence can be improved and the time taken by one iteration can be reduced. Therefore, we would like to explore some other technique such as quasi-Newton method, BFGS method, etc. The next idea we want to verify experimentally is if we can perform construction of just the fraction of the configuration space which is actually needed for the interpolation in real-time. If this approach turns out to be successful we address also some more advanced issues as topology changes.

The research has been supported by research intent “Integrated Approach to Education of PhD Students in the Area of Parallel and Distributed Systems” (No. 102/05/H050). We would also like to thank the authors of *Getfem* library. The implementation of the interaction was created in cooperation with the Robotics and Mechatronics Laboratory at Koç University, Istanbul.

References

1. Igor Peterlík and Luděk Matyska. An Algorithm of State-Space Precomputation Allowing Non-linear Haptic Deformation Modelling Using Finite Element Method. In *Second Joint EutoHaptics Conference and Symposium on Haptic Interfaces* Los Alamitos: IEEE Computer Society Press, 2007. pages 231–236
2. Evren Samur, Mert Sedef, Cagatay Basdogan, Levent Avtan and Oktay Duzgun. Robotic Indenter for Minimally Invasive Measurement and Characterization of Soft Tissue Behavior In *Medical Image Analysis*, 2007. Vol. 11, No.4, pages 361-373.
3. Stephane Cotin, Henry Delingette and Nicolas Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions On Visual. and Computer Graphics*, 5(1):62–73, 1999.
4. Guillaume Picinbono, Henry Delingette, and Nicolas Ayache. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *ICRA2001: IEEE International Conference Robotics and Automation*, Seoul Korea, May 2001.
5. Dan C. Popescu and Michael Compton. A model for efficient and accurate interaction with elastic objects in haptic virtual environments. In *GRAPHITE '03: Proceedings of the 1st internat. conf. on Computer graphics and interactive techniques*, pages 245–250,NY USA, 2003. ACM Press.
6. Yan Zhuang and John Canny. Real-time simulation of physically realistic global deformation. In *IEEE Vis'99*, 1999.
7. Xunlei Wu, Michael S. Downes, Tolga Goktekin, and Frank Tendick. Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes. In *EG 2001 Proceedings*, vol. 20(3), pages 349–358. Blackwell Publ., 2001.
8. Jernej Barbič and Doug L. James. Real-Time Subspace Integration of St.Venant-Kirchhoff Deformable Models in *ACM Transactions on Graphics (ACM SIGGRAPH 2005)*, 24(3), pages 982-990, August 2005.

Appendix C

Haptic Interaction with Soft Tissues Based on State-Space Approximation

by Igor Peterlík, Luděk Matyska

10 pages, accepted for Eurohaptics 2008 Conference, Madrid, to appear in Springer LNCS series

Haptic Interaction with Soft Tissues Based on State-Space Approximation

Igor Peterlík, Luděk Matyska

Faculty of Informatics, Masaryk University
Botanická 68a, Brno
Czech Republic
peterlik@ics.muni.cz, ludek@ics.muni.cz

Abstract. The well known property of haptic interaction is the high refresh rate of the haptic loop that is necessary for the stability of the interaction. Therefore, only simple computations can be done within the loop. On the other hand, physically-based deformation modeling requires heavy computations. Moreover, if realistic behavior is desired, the models are non-linear and iterative solution is necessary.
In this paper, the technique for haptic interaction with non-linear complex model is presented. Our approach, based on approximations of configuration space, is described and the accuracy of the approximation is experimentally determined.

Key words: haptic interaction, non-linear soft tissue modeling, interpolation

1 Motivation and Related Work

Recently, utilization of medical simulations has become an important and common part of the medical education. In particular, surgical simulators are of a great interest, as their importance for high-quality surgical training as well as operation planning still increases. In this context, the realistic interaction with soft tissues becomes an important area of research, as it represents the main issue in design of surgical simulators.

However, the demands for stable haptic interaction and realistic tissue behavior seem to be in contradiction, as it implies high-frequency haptic loop on one side, and computationally expensive solving in each iteration on the other. Moreover, according to the literature [1], the realistic models require non-linear properties, that implies iterative solving of large non-linear system of equations.

There have been several attempts to address this problem such as simplification of the underlying models or employing some precomputations before the real interaction occurs. Linear model together with haptic interaction is described in [2] applying *small area paradigm*. Non-linear model is proposed in [3] based on precomputation of special tensors, however the refresh rate is not sufficient and the degree of non-linearity is limited. The model with non-linear geometry is described in [4], several techniques as mass lumping are applied, however, again only the visual refresh rate is achieved. In [5] both geometrically and physically non-linear model is proposed, but no results about the speed and refresh rates are given. Recently, in [6] a new approach based on precomputation of polynomial coefficients for St.Venant material is presented. However, this approach is not applicable for other types of material.

The main contribution of this paper is to show that haptic interaction with complex deformable body can be realized also in the case, when non-linear models are considered. When comparing to other methods, the main advantage of our approach is that no assumptions about the complexity of the FEM mesh and underlying mathematical model are necessary, as all the expensive and iterative computations are done outside the haptic loop. In this paper, the proposed technique is described and its experimental validation is presented.

2 Deformation Modelling and Haptic Interaction

2.1 Physically Based Modelling

The main problem of physically based modeling can be informally formulated as follows: having a continuum body with defined boundary conditions, compute the deformation of the body in case when a force is applied either on the surface or in the volume of the body.

The main challenge is therefore to find a relation between the applied forces $\mathbf{f}$ and displacements $\mathbf{u}$ of the particles of the body. The relation is usually formulated within the frame of theory of elasticity, where the forces and displacements are internally represented by *stress and strain tensor*, respectively. The tensors are then coupled by constitution law that determines the behavior of the material. There are many various laws which are used for different kind of materials, e. g. St.Venant-Kirchhoff and Mooney-Rivlin which are suitable for soft tissues [7].

The constitution law is usually formulated as a partial differential equation. The most known method for solving this equation even on complex geometries is the *Finite Element Method.* It works with 3D mesh of the body and reformulates the problem into large system of algebraic equations $\mathbf{A}(\mathbf{u}) = \mathbf{f}$ where the forces and displacements in the nodes of the FEM mesh are computed. [8]. In the case when either full non-linear tensor or some non-linear constitution law is employed, the resulting algebraic system is non-linear and must be solved by some iterative method, e. g. combination of incremental loading and Newton method. For geometries having about thousand elements (considered in this paper), the assembly and solution process of linearized system performed in each iteration requires hundreds of millions floating point instructions.

2.2 Haptic Interaction with Deformable Body

In haptics, the deformation modeling concept can be slightly modified, as the input for the method is the prescribed displacement $\mathbf{t}$ of some part of the surface, which is in direct collision with haptic interaction point (HIP). As no other applied forces are considered, vector $\mathbf{f}$ is set to zero. Then, the task is to compute the reacting forces $\mathbf{h}$ applied on the HIP due to the collision. This modification is mathematically modeled by extension of the above mentioned system using *Lagrange multipliers*. Then, the augmented system $A(\mathbf{u}|\mathbf{h}) = 0|\mathbf{t}$ is solved.

In order to simplify the determination of the prescribed displacement vector $\mathbf{t}$, *point interaction* is considered. In this case the HIP is represented by single point. If a collision between HIP and mesh occurs, then the closest node of the mesh is selected as *active* and the HIP is attached to this node. From this point, the body is deformed by displacing the active node to some actual position $\mathbf{p}$ from its rest position $\mathbf{p}_0$. Therefore, the vector $\mathbf{t}$ consists of three components that are x, y, z displacement of the active node, i. e. $\mathbf{t} = \mathbf{p} - \mathbf{p}_0$.

In each iteration of the haptic loop, first the actual position of the active node is acquired resulting in the actual value of vector $\mathbf{t}$. Then, the reacting force $\mathbf{h}$ is computed and delivered back to the haptic device.

2.3 On-line and Off-line Computations

There are two basic approaches to the implementation of the haptic loop. In the first, the on-line computation of the reacting forces is done in each iteration of the haptic loop. This approach is usually used for linear models, when some small preparations are made immediately before the haptic loop is executed (e. g. the computation and decomposition of the system matrix which is constant during the interaction). The second approach assumes two separated phases of the interaction. First, extensive off-line precomputations are performed and the results are stored. Then, during the interaction, the precomputed data are used for calculation of the actual force feedback. In both cases, the number of floating-point instructions performed inside the haptic loop is reduced to hundreds or thousands.

As the first approach is not suitable when the matrices depend on the actual displacement, the precomputation scheme employing non-linear models is of great importance. In the following, we present the technique based on precomputation of configuration spaces that is suitable for highly non-linear models with complex geometry.

3 Configuration Space Approximation

3.1 Technique Based on Approximation of Discretized Spaces

In order to present the technique based on approximation of configuration space, first informal definitions of configuration and transition are introduced. The model composed of the soft body given by FEM mesh and HIP can be completely described by *configuration* C consisting of the actual position $\mathbf{p}$ of HIP, vector $\mathbf{u}$ of displacements of the FEM mesh nodes and force $\mathbf{h}$ acting in the

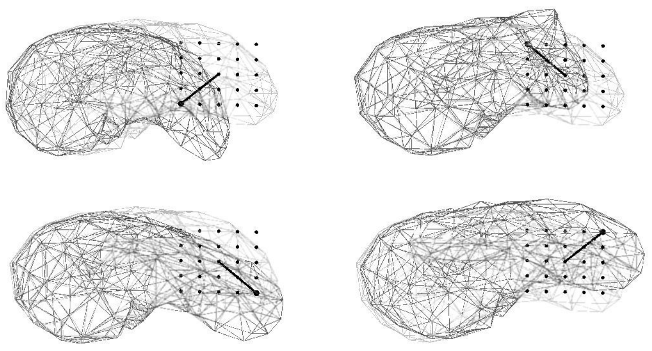

Fig. 1. Four configurations of the liver model. The active node is displaced to four distinct points of the grid (2D projection is made for simplification) and the forces and deformations are computed.

active node (attached to HIP). If the point interaction presented in the section 2.2 is considered, the configuration C is equivalent to the state vector of the interaction.

If the position of the active node is changed, i. e. $\mathbf{p} \rightarrow \mathbf{p}'$, then new reaction force $\mathbf{h}'$ and the deformation $\mathbf{u}'$ of the body are computed according to the underlying physical model, i. e. the *transition* $C \rightarrow C'$ between two distinct configurations occurs. Exploiting this concept, the haptic interaction can be regarded as travelling through configuration space when each step is given by a transition between two configurations. Having a fixed active node $\mathcal{N}$ the set of configurations $\mathcal{C}_{\mathcal{N}}$ constitutes a configuration space.

Before the formulation of the technique approximating an arbitrary configuration from discretized data, two assumptions are made:

A1 The configuration space is bounded, i. e. only configurations having the force amplitude $|\mathbf{h}|$ lower than some finite constant F_{max} are included in the space.
A2 The state-space is flat, i. e. each configuration C is uniquely determined by the position $\mathbf{p}$ of the active node.

The assumption A1 is quite natural, as there is limitation in force for each haptic device. The assumption A2 puts some restriction on the choice of the mathematical model, as any topological changes are not allowed. Nevertheless, there are neither restrictions about the size and complexity of the FEM mesh, nor assumptions about the non-linearities in the mathematical model. If the assumptions A1 and A2 hold, then following hypothesis can be formulated:

H1 There exists a discretization $\mathcal{D}_{\mathcal{N}} \subset \mathcal{C}_{\mathcal{N}}$ which is finite and can be effectively precomputed in the off-line phase of the interaction.
H2 Any configuration $C \in \mathcal{C}_{\mathcal{N}}$ can be approximated by some fast interpolation of the precomputed configurations in $\mathcal{D}_{\mathcal{N}}$. Moreover, the approximation can be computed during the on-line phase of the interaction.

Fig. 2. Discretization of the state space. The larger bullets represent precomputed configurations (compare with the figure 1). 0n the left only configurations in the grid points are depicted, whereas on the right, also intermediate states are included (represented by smaller bullets).

The statements H1 and H2 represent a technique which is suitable for haptic interaction with complex non-linear object, as the expensive computations are done in the first off-line phase, whereas during the interaction, the actual forces and deformations are computed by some fast interpolation of the precomputed data. Before the experimental validation of this hypothesis is presented in section 4, first precomputation phase is briefly described and second, possible implementations of the interpolation phase are given.

3.2 Precomputation Phase

According to the assumption A1, the configuration space is bounded due to the the force limitation. The straightforward construction of the discrete configuration space can be performed as follows: (i) the bounded space is discretized by a 3D regular grid p_{ijk}, (ii) the active node is displaced from its rest position to each point p_{ijk} of the regular grid and the corresponding configuration C_{ijk} is computed and inserted into $\mathcal{D}_{\mathcal{N}}$. A part of discretization is depicted by the Fig. 1 for four distinct configurations.

As the non-linear models are considered, the precomputation phase is computationally expensive. The efficient way how to construct $\mathcal{D}_{\mathcal{N}}$ is to compute transitions between the pairs of neighboring configurations $C_{i'j'k'} \rightarrow C_{ijk}$. The distance between neighboring configurations can be too large with respect to the convergence, so the incremental loading is applied: the transition $C_{i'j'k'} \rightarrow C_{ijk}$ is divided in M small steps and in each step, the intermediate configuration $C^m_{ijk}, m = 1 \ldots M$ is computed by iterative Newton method. In our implementation, M is fixed during the computations, however, in future it could be adjusted by the solver according the convergence of the calculations of the particular transition. The intermediate configurations C^m_{ijk} are also inserted into $\mathcal{D}_{\mathcal{N}}$. A part of the grid together with intermediate configurations is presented by the Fig. 2.

3.3 Interpolation Phase

In this part we focus on interpolation of the reacting force during the on-line phase. Each component (f_x, f_y, f_z) is interpolated from the precomputed forces separately. In the following, two different methods are considered: *polynomial* and *radial-based function* interpolation. For each of them, linear and cubic versions are employed.

Polynomial interpolation. As the polynomial interpolation works with regularly distributed data, only the configurations C_{ijk} stored in the points of the 3D grid are used. Having the actual position $\mathbf{p}_A$ of the active node displaced by HIP, the neighboring points $\mathbf{p}_{ijk}$ of the 3D grid must be first determined. For the simplest case of *trilinear interpolation*, eight neighboring nodes are needed for the interpolation, so the calculation is very fast. On the other hand, as smooth non-linear models are also considered the linear interpolation does not seem to be reliable.

The next alternative within the frame of the polynomial method is *tricubic interpolation*. In this case, 64 precomputed configurations are needed. The resulting tricubic form is $f(x,y,z) = \sum_{i,j,k=0}^{3} a_{ijk} x^i y^j z^k$. where the coefficients a_{ijk} can be calculated from the precomputed forces within the off-line phase of the interaction.

Due to the non-linearity of the underlying problem, the convergence for some configurations does not have to be achieved during the off-line phase. In that case, the resulting configuration is not completely valid, as the forces and displacements are not computed with the desired accuracy. However, as the polynomial interpolation requires the regular grid, the non-convergent configurations cannot be excluded from the discretization $\mathcal{D}_{\mathcal{N}}$ and they can seriously affect the accuracy of the interpolation.

Radial-based function Interpolation. According to the text above, there are two reasons why a method interpolating from *scattered (irregularly distributed) data* is desired:

- both the configurations C_{ijk} in the points of the regular grid and C^m_{ijk} in the intermediate states can be utilized
- before the interpolation takes place, the configurations computed in non-convergent iterations can be excluded from $\mathcal{D}_{\mathcal{N}}$

To meet these requirements, *radial-based function* (RDF) interpolation was employed. Briefly, having a set of values $f(\mathbf{x}_j)$ for positions $\mathbf{x}_j$ placed irregularly in space, then arbitrary value of $f(\mathbf{x})$ for a position $\mathbf{x} = (x, y, z)$ is calculated as:

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} w_i \phi(|\mathbf{x} - \mathbf{x}_i|) \tag{1}$$

where ϕ is a chosen function, e. g. $\phi(r) = r$ or $\phi(r) = r^3$ determining linear and cubic interpolation, respectively.

Further, there are weights w_i that can be computed in the off-line phase of the interaction by solving and decomposing a linear system which is assembled by substituting the known values $f(\mathbf{x}_j)$ into the Eq.1. Having the weights precomputed, the interpolation for the actual position can be computed quickly even for large number of interpolation points.

As data can be scattered in the configuration space, an arbitrary subset of the precomputation configurations can be used in order to increase the efficiency and accuracy of the interpolation.

4 Results and Discussion

4.1 Methodology of Validation

As stated in section 2.2 the non-linear model cannot be computed in real-time during on-line phase of the interaction. In order to validate the approach based on the interpolation of configurations, a set of completely random paths in the configuration space $\mathcal{C}_\mathcal{N}$ were generated. More precisely, having an active node $\mathcal{N}$, sequences of configurations $C_1 \to C_2 \to \ldots \to C_N$ were constructed in small steps using the Newton method for computation of each transition $C_i \to C_{i+1}$. This process can be regarded as an "simulation" of the haptic interaction, when the actual configuration is iteratively computed in each step.

Let $\mathcal{S}_\mathcal{N}$ be the set of all the configurations, computed precisely in the above process of simulation. Then, for each $C \in \mathcal{S}_\mathcal{N}$ the counterpart $\hat{C}$ is calculated using the interpolation methods described in the section 3.3.

To evaluate the accuracy of the interpolation for a particular active node $\mathcal{N}$, mean relative error $\bar{r}_\mathcal{N}$ and mean absolute error $\bar{a}_\mathcal{N}$ of force interpolation for an active node $\mathcal{N}$ are defined:

$$\bar{r}_\mathcal{N} = \frac{1}{|\mathcal{C}_\mathcal{N}|} \sum_{C \in \mathcal{S}_\mathcal{N}} \frac{\|\mathbf{h}_C - \hat{\mathbf{h}}_{\hat{C}}\|}{\|\mathbf{h}_C\|} \qquad \bar{a}_\mathcal{N} = \frac{1}{|\mathcal{C}_\mathcal{N}|} \sum_{C \in \mathcal{S}_\mathcal{N}} \|\mathbf{h}_C - \hat{\mathbf{h}}_{\hat{C}}\|, \tag{2}$$

where $\mathbf{h}_C$ is the reacting force in active node for the precise configuration C and $\hat{\mathbf{h}}_{\hat{C}}$ is force vector corresponding to the interpolated counterpart $\hat{C}$.

Besides the mean errors, the values of maximal relative error $R_\mathcal{N}$ and maximal absolute error $A_\mathcal{N}$ were recorded during the validation, in order to determine the worst case that occurred during the validation. Both maximal errors were computed analogically as in Eq. 2 where the averaging function was replaced by the maximization.

4.2 Implementation and Results

For the implementation purposes, the linear St.Venant-Kirchhoff material was used with material coefficients derived in [7]. However, as realistic modeling of large deformation is desired, Green strain tensor was used, resulting in non-linear mathematical model. The assembly procedures and iterative solution of the system were implemented using the C++ library GetFEM[9]. For the experiments, 3D finite element of mesh of pig liver with 281 nodes was used. The second order FEM was used resulting in more than 5400 degrees of freedom.

The configuration spaces for ten various active nodes were precomputed. Each configuration space was discretized by regular grid consisting of $7 \times 7 \times 7$ points and the maximal reacting force $F_{max} = 30\,N$ was achieved. Within the computation of transition to each configuration stored in some point of the mesh, six intermediate states were computed. In total, 343 configurations were placed in grid points and about 1700 of configurations were computed within the intermediate states. Due to convergence issues, approximately 10% of the precomputed configurations were not valid as the numerical method did not converge with desired precision. The precomputation of the configuration spaces were done in distributed environment and it took approximately 50 hours being executed on ten 3 GHz Xeon processors. It should be noted that the total time of computations will be reduced in future by employing a solver more suitable for calculations of deformations.

For the validation purposes, the simulation of random paths described in the previous section was computed. For each of the selected active nodes, between 1500 and 3500 configurations were generated giving more than 25000 randomly placed configurations in total. The simulation was also computed in parallel, taking about 70 hours on 30 CPUs. The interpolation of the precomputed data together with the statistical evaluation were implemented in Matlab environment. The Table 1 presents the mean and maximal errors that were recorded during the experimental validation process.

		Polynomial				Radial-Based Function			
		Trilinear		Tricubic		Linear		Cubic	
n	$\lvert\mathcal{S}_{\mathcal{N}}\rvert$	$\bar{r}_{\mathcal{N}}$ [%]	$\bar{a}_{\mathcal{N}}$ [N]	$\bar{r}_{\mathcal{N}}$ [%]	$\bar{a}_{\mathcal{N}}$ [N]	$\bar{r}_{\mathcal{N}}$ [%]	$\bar{a}_{\mathcal{N}}$ [N]	$\bar{r}_{\mathcal{N}}$ [%]	$\bar{a}_{\mathcal{N}}$ [N]
5	3448	8.9	0.27	1.41	0.027	1.28	0.045	0.29	0.007
13	4159	11.7	0.52	1.75	0.059	1.39	0.119	0.40	0.024
14	3480	12.2	0.50	1.06	0.054	0.46	0.039	0.21	0.012
16	1697	14.2	0.19	3.27	0.021	0.70	0.014	0.26	0.003
78	2437	12.5	0.40	2.50	0.053	1.17	0.094	0.33	0.018
81	1571	13.5	0.21	2.59	0.030	1.38	0.042	0.48	0.012
87	1565	10.1	0.43	1.01	0.050	0.98	0.058	0.36	0.020
133	3035	9.4	0.41	1.44	0.047	0.74	0.038	0.34	0.015
181	2183	10.1	0.64	3.12	0.066	1.27	0.159	0.36	0.031
230	1560	9.4	0.39	1.22	0.044	1.31	0.077	0.45	0.026

		Polynomial				Radial-based Function			
		Trilinear		Tricubic		Linear		Cubic	
n	$\lvert\mathcal{S}_n\rvert$	$R_{\mathcal{N}}$[%]	$A_{\mathcal{N}}$[N]	$R_{\mathcal{N}}$[%]	$A_{\mathcal{N}}$[N]	$R_{\mathcal{N}}$[%]	$A_{\mathcal{N}}$[N]	$R_{\mathcal{N}}$[%]	$A_{\mathcal{N}}$[N]
5	3448	85	1.1	43	0.56	22	0.73	13.2	0.34
13	4159	404	1.7	68	0.76	32	3.08	12.6	0.48
14	3480	446	2.6	16	0.66	11	2.49	10.2	0.37
16	1697	198	0.6	63	0.34	21	0.34	8.3	0.07
78	2437	317	7.2	287	6.55	28	3.16	7.1	0.47
81	1571	395	0.5	112	0.29	22	0.78	4.2	0.15
87	1565	284	1.6	8	0.29	22	1.13	5.4	0.20
133	3035	108	1.9	29	0.91	15	0.57	5.7	0.24
181	2183	208	2.7	131	0.37	38	4.01	16.0	0.71
230	1560	152	1.0	28	0.69	24	1.26	5.1	0.40

Table 1. Mean errors (the upper table) and maximal errors (lower table) for ten various active nodes and eight types of interpolation; $\mathcal{N}$ denotes the index of the node and $|\mathcal{S}_{\mathcal{N}}|$ the number of configuration computed for the particular active node.

4.3 Discussion

In this section, the results presented in Table 1 are discussed. First, when comparing four types of interpolation presented in 3.3 w. r. t. the mean errors, following observations are made:

- The trilinear polynomial interpolation shows the worst results according to the expectations stated in the section 3.3. The polynomial tricubic interpolation gives reasonable results; although the mean relative errors exceeds 3 %, the mean absolute error stays bellow $0.1\,N$.
- The radial-based function interpolation shows very good results even for the linear version, resulting in lower both relative and absolute errors than tricubic interpolation.
- Finally, the cubic RBF gives the best results, as the mean differences between the precise and interpolated values are negligible.

However, the optimistic results suggested by mean errors are overshadowed by the maximal errors showing the superiority of RBF interpolation:

- The trilinear polynomial interpolation turns out to be completely unacceptable, since for some special cases, the relative error approaches 500% and moreover, the absolute error is certainly not negligible (even 7 N). However, the tricubic interpolation does not give reasonable results as well as in some cases, the error exceeds 200 % (6 N).
- The bad results achieved by polynomial interpolation are caused by the fact, that only a fraction of the precomputed configurations are used including the configurations computed by non-convergent solution. Although the mean error is quite satisfactory, there is a number of configurations that were interpolated from erroneous data and in this case, the interpolated values are completely invalid.
- The linear RBF turns out to be less reliable, too. Although, all the convergent states were used, the interpolation is not smooth and accurate enough to approximate the non-linear behavior.
- The only acceptable results were achieved with the cubic RBF function. For some cases, the maximal error exceeds 10 %, however, the amplitude of the error almost never exceeds the value of 0.5 N and the values of mean errors indicate that the number of states with error exceeds 1% (or 0.1 N) is low.

The results and discussion above experimentally confirm the hypothesis from the section 3.1. The construction of the desired discretization $\mathcal{D}_{\mathcal{N}}$ described in section 3.2 can be done in finite time as summarized in section 4.2. Further, the configurations stored in discretization $\mathcal{D}_{\mathcal{N}}$ can be used for approximation of a configuration for arbitrary position of HIP using some interpolation method from the section 3.3 which is fast enough to be computed in haptic loop. Finally, the Table 1 shows there is an acceptable upper bound limit for the approximation error assuming that cubic RBF interpolation is utilized.

This implies that the real-time haptic interaction with complex non-linear model is possible based on the approach employing the precomputations of configuration space and interpolation of the precomputed configurations.

5 Conclusion and Acknowledgement

The technique based on the approximation of configuration spaces allowing haptic interaction with complex body with non-linear behavior has been described. Experimental validation of the technique has been presented based on comparison of several interpolation methods. The method was employed for implementation of haptic interaction by Robotics and Mechatronics Laboratory at Koç University, Istanbul. The StVenant-Kirchhoff and Mooney-Rivlin incompressible materials with non-linear strain tensor were employed and finite element mesh of human liver consisting of 10-point tetrahedra was utilized resulting in 5400 non-linear equations. The refresh rate over 1 kHz was achieved when calculating the polynomial interpolation of precomputed data inside the haptic loop.

In the future, we would like to first focus on the optimization of the solver to reduce the precomputation time by improving the convergence of the calculations.. Further, we would like to employ dynamical properties of the tissues.

The implementation of the haptic interaction was created in cooperation with the Robotics and Mechatronics Laboratory at Koç University, Istanbul, who also provided the data. The research has been supported by Ministry of Education, Youth and Sport of the Czech Republic under the research intent no. 102/05/H050. We also thank to the authors of the GetFEM library.

References

1. S.Misra, A.Okamura and K.T.Ramesh. Force Feedback is Noticeably Different for Linear versus Nonlinear Elastic Tissue Models Proceedings of the *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces*, 2007,519–524.
2. D.C.Popescu and M.Compton. A model for efficient and accurate interaction with elastic objects in haptic virtual environments. In *GRAPHITE '03: Proceedings of the 1st internat. conf. on Computer graphics and interactive techniques* 2003,245–250.
3. G.Picinbono, H.Delingette and N.Ayache. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *ICRA2001: IEEE Int. Conference Robotics and Automation* 2001
4. Y.Zhuang and J.Canny. Real-time simulation of physically realistic global deformation. In *IEEE Vis'99*, 1999.
5. X.Wu, M.S.Downes, T.Goktekin and F.Tendick. Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes. In *EG 2001 Proceedings*, 2001, 20(3):349–358.
6. J.Barbič and D.L.James. Real-Time Subspace Integration of St.Venant-Kirchhoff Deformable Models In *ACM Transactions on Graphics (ACM SIGGRAPH 2005)*, 2005, 24(3):982–990.
7. E.Samur, M.Sedef, C.Basdogan, L.Avtan and O.Duzgun. Robotic Indenter for Minimally Invasive Measurement and Characterization of Soft Tissue Behavior In *Medical Image Analysis* 2007, 11(4):361–373.
8. M.Bro-Nielsen. Finite element modeling in surgery simulation Proceedings of the IEEE 1998, 86(3):490–503.
9. J.Pommier and Y. Renard. *Getfem++, an open source generic C++ library for FEM*, http://www-gmm.insa-toulouse.fr/getfem.