17

Haptic Interaction with Complex Models Based on Precomputations

Igor Peterlík and Luděk Matyska and Jiří Filipovič
Masaryk University
Czech Republic

1. Introduction

The real-time haptic interaction with deformable objects is an important area of research with wide range of applications in medicine and industry. The development of computer-based medical training systems (medical simulators) is perhaps the most challenging task, as it requires physically-based modelling to mimic the behaviour of soft tissues. Such models are usually based on the mathematical formulations emerging from the theory of elasticity, resulting in non-linear boundary-value problems defined over complex domains. Numerical techniques such as the finite element method, which are needed to solve these problems, are computationally expensive and therefore, their employment in the real-time interaction is not straightforward, especially in haptics, where the refresh rate over 1 kHz is required.

There have been several approaches proposed so far to address the issue of coupling the heavy computations with high refresh rate of the haptic loop. Basically, two main classes of solutions can be identified: either simplified models based on linearization or reduction are used, or some precomputation is employed before the real-time interaction takes place. In the first part of this chapter, an overview of the methods proposed in last decade is presented. Some of them are described in detail to emphasize the key concepts which are used frequently in haptic soft-tissue modeling nowadays.

In the second part of the chapter, an approach based on precomputation and interpolation of precomputed data is presented. The technique consists of two procedures: first, it is a construction of a discrete set of data, which can be performed in reasonable time on today's computers. Second, it is a fast approximation of an arbitrary deformation which is needed during the real-time interaction from the data constructed by the first procedure.

In the text, both procedures are described conceptually emphasizing the computational aspect. An algorithm of distributed state-space search used for the precomputation procedure is presented and then, three different types of the interpolations are studied and compared w. r. t. the main features which are employed in the real-time interaction phase. After general description, the evaluation of the method is briefly given using a set of experiments being done with a finite element 3D model of liver.

Finally, possibilities to couple the precomputation with interpolation, leading to a system capable of real-time interaction without off-line computations, is shortly discussed. The chapter is concluded with summary of main features and advantages of the presented systems for haptic interaction with complex models.

2. Methods for Haptic Rendering of Deformable Objects

2.1 Overview of Modelling Methods

The methods in the area of the deformation modelling can be ranged into two main groups which are denoted as non-physical and physical. The **non-physical methods** are used mainly in the computer graphics, as they are very fast and efficient. Two well-known representatives are spline modelling and free form deformations. In **spline modelling**, the curves and surfaces are represented by a set of control points and the shape of the objects is modified by changing the position of the points. The main idea of **free form deformations** is to deform the shape of the object by deforming the space in which the object is embedded. Generally, the non-physical methods are not suitable in the case when the physically realistic behaviour of the deformations is desired. In this case, the physical modeling is the only alternative.
The **physical methods** are based on the mathematical models usually formulated by *partial-differential equations* (PDE). However, the main issue of this method is represented by the fact that the resulting problem formulation is complex and the analytical solution is computationally expensive or even infeasible. To address this issue, the models are simplified to cover the essential observations and the equations are solved numerically. Bellow, the main methods used in the physical modelling are presented with a brief description.

Mass-spring damper method (MSD). The mass is concentrated in a number of nodes which are connected by springs, usually modelled as linear. When a force is applied to a node, it starts to move and pass the force via springs to the neighbouring nodes. From the computational point of view, the method is very simple and the cost of the calculation is low. Although the method is based on physical model, the major drawback of MSD systems is the insufficient approximation of real material properties. They also suffer from inaccuracy of the approximation in the case when the geometry of the object is complex.

Finite difference method (FDM). The continuous derivative which appears in the PDE-based formulations is replaced with a finite difference approximation, which is computed in points organized in regular grid which spans over the domain of the object. The technique is very accurate and efficient for the objects with regular geometry. Nevertheless, in case of complex shape of the domain, the discretization becomes extremely dense, resulting in a huge computational complexity.

Boundary element method (BEM). 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. The models based on this model cannot cope with any phenomena related to the volume of the body because of the reduction presented above, so e. g. the applied volume forces or heterogeneity of the body cannot be considered.

Finite element method (FEM). 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 discretization of the domain and mathematical re-formulation of the boundary-value problem resulting in large systems of algebraic equations. The finite element method seems to be superior to the other methods when the modelling complex bodies with non-trivial physical properties is considered. It is also suitable when volumetric operations such as cutting and tearing are to be modelled, as the entire domain of the body is included in the formulation.

(a) (b)

Fig. 1. Deformations of FE mesh of human liver (rest position depicted in gray) for (a) linear elasticity model, (b) non-linear Mooney-Rivlin model with complete strain tensor. In both cases, the same force was applied to a node in the left part of the body.

In the following text, we focus on the methods which are based on the physical approach. More precisely, the finite element (FE) models are briefly introduced and the most important methods employing the real-time haptic interaction are surveyed.

2.2 Overview of FE models

The physically based deformation modeling is based on relation derived within the frame of elasticity theory. A detailed description of the mathematical formulation can be found for example in Ciarlet (1988); J.T.Oden (1972); Wriggers (2008). The realistic behaviour of the objects modelled by the finite element method is usually validated w. r. t. the real measurements. In the following text, only some basic terms are informally introduced in order to identify the main issues which are associated with real-time haptic modeling of soft tissues.

When speaking about the modeling of deformations, two types of non-linearities are usually considered. First, the **geometric non-linearity** introduces non-linear relation between the displacement and strain. In case when the non-linear term in the definition of the strain tensor is neglected, only small deformations are rendered correctly. For larger deformations, the volume of the deformable object is not preserved and the overall behaviour of the object is not realistic. Therefore, geometrically non-linear model must be used when large deformations take place. The difference in behavior between geometrically linear and non-linear model is demonstrated in Fig. 1.

Second, **physical non-linearity** is introduced if non-linear relation between the stress and strain is used. The visual difference between physically linear and non-linear model is not so apparent, nevertheless, this type of non-linearity heavily affects the force response of the body and it plays an important role for realistic simulation of the soft tissues (see Misra et al. (2007)). Physically non-linear model is represented by Mooney-Rivlin material employing the non-linear incompressibility condition.

Putting it all together, both non-linearities are of a great importance when realistic modelling is required, since the geometrically linear model cannot handle large deformations properly, whereas physically linear model is limiting w. r. t. the material properties. Nevertheless, both geometrically and physically linear models have been extensively used in the past because of

lower computational cost, so they represent the first efforts in the area of haptic deformation modeling. Also, the linear models turn out to be sufficient for modeling of deformations in various laparoscopic simulators where mostly the small deformations are encountered.
Therefore, the survey of methods being proposed for the haptic deformation modeling aims at linear models first. Then, techniques proposed to combine the computationally demanding non-linear models with real-time haptic interactions are presented. In both cases, the methods employing various types of precomputations are emphasized, nevertheless some other important approaches are mentioned as well.

2.3 Linear Models

The pioneering approach in the area of soft tissue modelling is presented in Bro-Nielsen & Cotin (1996). Several important concepts proposed there have been successfully re-used since then. First, the model is completely based on physical formulation of deformations. The finite element method is used to reformulate the relations from the theory of elasticity, using the discretization of the domain by 3D tetrahedral mesh and approximation of the solution with linear shape functions over the elements, arriving to the system of linear equations

$$\mathbf{K}\mathbf{u} = \mathbf{f} \tag{1}$$

where $\mathbf{u}$ represents the vector of nodal displacements, $\mathbf{f}$ denotes the external forces (and applied loads) and $\mathbf{K}$ is a linear mapping between the two quantities. In this case, $\mathbf{K}$ is given by a $3n \times 3n$ matrix where n is the number of nodes in the FE mesh. It is important to realize that in the case of linearity of the system 1, the matrix $\mathbf{K}$ is invariant w. r. t. the displacement $\mathbf{u}$. The simulation is **driven by forces**, i. e. the force is regarded as the input and finite element method is used to compute the deformations.
The main goal of the proposal is to increase the speed of the simulation as much as possible at the cost of time consuming precomputation phase performed before the simulation, memory requirements and lower accuracy of the simulation. Three concepts are proposed to increase the speed of the computations shifting it towards the real-time scene. First, a concept of **condensation** is applied to the system: the nodes of the mesh are divided into *surface* and *internal* and the linear system is reordered as follows:

$$\begin{bmatrix} \mathbf{K}_{ss} & \mathbf{K}_{si} \\ \mathbf{K}_{is} & \mathbf{K}_{ii} \end{bmatrix} \begin{bmatrix} \mathbf{u}_s \\ \mathbf{u}_i \end{bmatrix} = \begin{bmatrix} \mathbf{f}_s \\ \mathbf{f}_i \end{bmatrix}. \tag{2}$$

Using the blocked system above, $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{f}}$ are defined as

$$\tilde{\mathbf{K}}_{ss} = \mathbf{K}_{ss} - \mathbf{K}_{si}\mathbf{K}_{ii}^{-1}\mathbf{K}_{is} \qquad \tilde{\mathbf{f}}_s = \mathbf{f}_s - \mathbf{K}_{si}\mathbf{K}_{ii}^{-1}\mathbf{f}_i \tag{3}$$

and a reduced system $\tilde{\mathbf{K}}_{ss}\mathbf{u}_s = \tilde{\mathbf{f}}_s$ involving only the m nodes is solved during the simulation, so that $\tilde{\mathbf{K}}_{ss}$ is $3m \times 3m$ matrix for $m \ll n$. It is emphasized that although the size of the system is given only by the surface nodes, the behaviour of the system is the same as in the case of the volumetric FE model. Besides the static scenario, the dynamic system is considered as well employing *mass matrix* $\mathbf{M}$ and damping matrix $\mathbf{D}$:

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f}. \tag{4}$$

It is shown that the condensation can be used also for the dynamic system in combination with the finite difference scheme applied for time integration.

Second, another improvement of the method is given by the fact that the reduced linear system is not solved by some conventional iterative method such as Conjugate Gradients, but **direct inversion** of the stiffness matrix is used. And this is the place where the precomputations are employed: since the matrix is constant, the inverse $\mathbf{K}^{-1}$ can be precomputed and stored explicitly before the interaction starts. The authors admit that the direct inversion can introduce numerical errors and moreover, the resulting inverse is no more a sparse matrix. Nevertheless, the trade-off between the precomputation overhead, memory requirements and accuracy on one side and speed of the computations on the other is acceptable. Third, a concept of **selective matrix vector multiplication** is employed: since the force vector $\mathbf{f}$ contains only a small number of non-zero elements, the product $\mathbf{K}^{-1}\mathbf{f}$ is done selectively, i. e. only for the non-zero positions of the vector $\mathbf{f}$. It is shown however, that this approach can be applied only in the case of static model, since in the dynamic case, the corresponding force vector is not sparse any more due to the finite difference scheme.
Putting it all together, the computation of deformations based on static finite element model using the condensation, precomputation of direct inverse of the stiffness matrix and selective multiplication is denoted as **fast finite elements**. The method is used for the visual simulation of human leg with 700 surface nodes (as the number of internal nodes does not affect the speed of the simulation) achieving 20 frames per second. Further, the authors proposed a parallelized version of the technique based on domain decomposition. However, the parallel implementation employing iterative solution method (preconditioned CG) turned out to be too slow to achieve a real-time performance. Another extension proposed by the author in (Bro-Nielsen, 1996) is represented by cutting in finite element system which is performed by removing an element from the FE mesh. Such a modification requires precomputation of the stiffness matrix inverse which is computationally demanding and cannot be done in real-time. Nevertheless, much more efficient solution is proposed by the authors based on Woodbury formula as shown in (Hager, 1989).
The concept of condensation of linear FEM introduced above is used also in (Frank et al., 2001). The nodes are divided into two categories — those with given boundary conditions and other nodes which are not directly involved in the interaction. A cube model composed of linear tetrahedra is used for testing: besides a mesh with 375 DOFs, meshes with higher resolution are used, both increasing the order of the elements (*p-resolution*) and decreasing the element size (*h-resolution*). The reduced system is solved by direct (Cholesky) and iterative methods (Conjugate Gradients, Preconditioned-Jacobi and Preconditioned SSOR). It is shown that only for the coarsest model, the haptic refresh rate (500 Hz) can be achieved using the Conjugate Gradients method. Beside the conventional solvers, full multi-grid method is used exploring the hierarchy of meshes.
The method based on condensation and precomputation of the inverse of the stiffness matrix is also used in work published in (Gosline et al., 2004). The simulation of the fluid pockets inside a soft tissue is developed using static linear elasticity and static fluid analysis. The speed of the simulation is improved significantly by condensation and off-line precomputation of the inverse matrices needed for solving of the reduced system. The needle insertion is then simulated on a single machine using a model with 82 surface nodes achieving refresh rate of more than 500 Hz.
The condensation technique is also explored in (Chai et al., 2001) where a step towards nonlinear models is proposed. The soft tissue is divided into two parts: first, a small area that is in direct contact with operation tool is modeled more accurately, so the stiffness matrix corresponding to the nodes from the area is updated during the interaction. The rest of the

body is considered as linear, so the corresponding part of the stiffness matrix is constant and the condensation and inverse precomputation can be applied. The authors state that real-time refresh rate is achieved in their setting for a body with 6601, including 861 nodes in the "operational part" of the body, however, no further quantitative results are provided.
The combination of precomputation and condensation is proposed in Nikitin et al. (2002). In the off-line mode, the global stiffness matrix K was assembled and further, the block $(K)_S$ with rows corresponding to the visible surface nodes was inverted obtaining $(K^{-1})_S$. By this simplification, time as well as storage were reduced significantly.
A novel approach to soft tissue modelling was proposed in Cotin et al. (1996). The model introduced in the paper is based on linear elasticity, so small deformations are modelled realistically. The equations are reformulated using the finite element method; the domain of the body is discretized by tetrahedral FE mesh. The simulation is proposed as being **driven by displacements**: the displacements of the fixed nodes are given as input (so called applied loads) whereas the response forces are computed by the FEM together with the overall deformation. Mathematically, the method of Lagrange multipliers is used to obtain the reaction forces corresponding to the applied loads as shown in (Cotin et al., 1999). The method results in augmented system $\hat{\mathbf{K}}\hat{\mathbf{u}} = \hat{\mathbf{f}}$ defined as

$$\begin{bmatrix} \mathbf{K} & \mathbf{A} \\ \mathbf{A}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{u}^* \end{bmatrix} \tag{5}$$

where $\mathbf{u}^*$ is a vector of pseudo-loads given by the prescribed displacement of touched nodes and $\mathbf{A}$ is a rectangular constraint sparse matrix containing non-zero elements only at the positions corresponding to the nodes with prescribed displacements. Further, the vector $\boldsymbol{\lambda}$ represents the Lagrange multipliers that is obtained by the solution of the system. The method is suitable for the haptic interaction using impedance-control devices, as the multipliers $\boldsymbol{\lambda}$ directly represent the response force associated with the pseudo-loads given by $\mathbf{u}^*$.
In order to speed up the rate, the linearity of the model is explored resulting in **superposition principle**. For each free surface node k that is not fixed by a boundary condition, an elementary displacement is applied and two quantities are precomputed: first, for each other free node $n \neq k$ the displacement induced by the applied load in node k is computed and stored as tensor $[\mathbf{T}^{\mathbf{u}}_{\mathbf{nk}}]$. Second, elementary response force at node k is calculated and stored as tensor $[\mathbf{T}^{\mathbf{f}}_{\mathbf{k}}]$. All the tensors are computed by solving the linear system using the conjugated gradient method. During the interaction, both the actual deformation of the body and reaction force are calculated from the precomputed tensors using the superposition principle, i. e. the deformation in node n and force in node k are computed as

$$\mathbf{u}_n = \sum_{l=1}^{M} [\mathbf{T}^{\mathbf{u}}_{\mathbf{nl}}][\mathbf{u}^*_l] \qquad \mathbf{f}_k = [\mathbf{T}^{\mathbf{f}}_{\mathbf{k}}][\mathbf{u}^*_k] \tag{6}$$

where $\mathbf{u}^*_k$ is corrected applied load computed from the actual prescribed displacement of the node k. Computing time of 7 ms is reported using the superposition principle for a model with 1400 nodes and 6500 tetrahedra after 8 hours of precomputations of the tensors corresponding to the elementary displacements.
The model presented above is based on static equilibrium and the simulation of the dynamic behaviour is replaced by a **quasi-static** approach: the deformation at time t is computed as static solution of the boundary-value problem utilizing the boundary conditions imposed in time $t-1$. Due to the precomputation of the tensors, any modifications of mesh topology (e. g.

tearing or suturing) are not possible. Therefore in Cotin et al. (2000a), a dynamic **tensor-mass** model is proposed allowing for topology changes. In this case each tetrahedron is associated with data structure containing tensors corresponding to its vertices and edges. The mass and damping matrices in the dynamic Eq. 4 are considered as diagonal (simplification known as **mass lumping** that allows to decouple the motion of the nodes, so the dynamic Eq. 4 can be integrated separately for each node of the FE mesh.
The fourth-order Runge-Kutta integration method is used to solve the dynamic equation in each step of the simulation. The actual force is calculated from the tensors, which are used to determine the local stiffness between the adjacent tetrahedra. The tensors are also computed separately for each tetrahedron directly during the haptic loop. The cutting and tearing is implemented by tetrahedra removal. In this case, only the tensors corresponding to the adjacent elements are updated. The refresh rate of 40 Hz for mesh with 760 vertices and 4000 edges is reported on computer equipped with 233 MHz Alpha processor.
Since the mass-tensor model is computationally more expensive than the previous one based on the precomputations, **hybrid approach** combining both is proposed in Cotin et al. (2000b). The part of the tissue where the operation takes place is modelled by the dynamic mass-tensor model allowing cutting, whereas the rest of the body is modeled by the quasi-static model based on the precomputation of the elementary displacements. The model is further extended in Picinbono et al. (2002) by introducing anisotropic properties To achieve the haptic refresh rate, **force extrapolation** is proposed when the forces updated at the frequency of 500 Hz are generated from the simulation based on the hybrid model running at 30 Hz. Linear extrapolation is employed, so the force estimation $\mathbf{f}^e(t)$ at time t, $t_n \leq t < t_{n+1}$ is computed as

$$\mathbf{f}^e(t) = \mathbf{f}_n + \frac{t - t_n}{t_n - t_{n-1}}(\mathbf{f}_n - \mathbf{f}_{n-1}). \tag{7}$$

In Delingette & Ayache (2005), the methods based on hybrid model and further extensions are summarized and the successful implementation of hepatic surgery simulator **Laparoscopic Impulse Engine** is reported.
In (James & Pai, 2002), a unified treatment of the elastostatic simulation is proposed based on Green functions. It is shown that any deformation simulated by the linear model can be computed using the Green functions as a basis. Algorithm for precomputation of the Green function based on capacitance matrices is presented and example for boundary elements is derived.
A model based on linear static elasticity is presented in Popescu & Compton (2003). To achieve the high refresh rate needed for the real-time haptic interaction, **small-area paradigm** is introduced, being an extension of the displacement-driven interaction based on the solution of the augmented system $\hat{\mathbf{K}}\hat{\mathbf{u}} = \hat{\mathbf{f}}$ defined by 5. It is assumed that the area affected by the haptic device is small, i. e. the vector of prescribed displacements $\mathbf{u}^*$ has only m elements where m is a small number. Moreover, no external forces are applied on the tissue, so the vector $\mathbf{f}$ is zero. Since linear elasticity is considered, the stiffness matrix $\mathbf{K}$ is constant (as well as its inverse) and only the constraint matrix $\mathbf{A}$ varies within the augmented matrix $\hat{\mathbf{K}}$ during the interaction. However, the system 5 can be inverted as

$$\begin{bmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \mathbf{A} \\ \mathbf{A}^\top & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f} \\ \mathbf{u}^* \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{P}} & \tilde{\mathbf{Q}} \\ \tilde{\mathbf{R}} & \tilde{\mathbf{S}} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{u}^* \end{bmatrix} \tag{8}$$

where

$$\tilde{\mathbf{P}} = \mathbf{K}^{-1} - (\mathbf{K}^{-1}\mathbf{A})(\mathbf{A}^{\top}\mathbf{K}^{-1}\mathbf{A})^{-1}(\mathbf{A}^{\top}\mathbf{K}^{-1}) \quad (9)$$

$$\tilde{\mathbf{Q}} = (\mathbf{K}^{-1}\mathbf{A})(\mathbf{A}^{\top}\mathbf{K}^{-1}\mathbf{A})^{-1} \quad (10)$$

$$\tilde{\mathbf{R}} = (\mathbf{A}^{\top}\mathbf{K}^{-1}\mathbf{A})^{-1}(\mathbf{A}^{\top}\mathbf{K}^{-1}) \quad (11)$$

$$\tilde{\mathbf{S}} = (\mathbf{A}^{\top}\mathbf{K}^{-1}\mathbf{A})^{-1}. \quad (12)$$

Having the vector $\mathbf{u}^*$ of the prescribed displacements, the corresponding force response stored in $\boldsymbol{\lambda}$ is computed as $\boldsymbol{\lambda} = \tilde{\mathbf{S}}\mathbf{u}^*$. The matrix $\tilde{\mathbf{S}}$ can be computed in real-time: using the relation 12, a submatrix $\tilde{\mathbf{S}}^{-1} = \mathbf{A}^{\top}\mathbf{K}^{-1}\mathbf{A}$ of $\mathbf{K}^{-1}$ is extracted. The small area paradigm implies that $\tilde{\mathbf{S}}^{-1}$ is a small $m \times m$ matrix that can be inverted easily by some direct method to obtain $\tilde{\mathbf{S}}$.
The computational framework is set as follows: first, the sparse matrix $\mathbf{K}$ is assembled using finite element method and its inverse $\mathbf{K}^{-1}$ is computed and stored. During the interaction, the reaction force $\boldsymbol{\lambda}$ is computed on high-frequency by inverting the small submatrix of $\mathbf{K}^{-1}$ and multiplying by $\mathbf{u}^*$. The overall deformation is computed by analogy using the definition of the matrix $\tilde{\mathbf{Q}}$. The implementation of the model showed good results: using Silicon Graphics Onyx2 with 8 CPU running at 250 MHz, the computation of the mesh deformation was 0.0277 s and 0.0493 s respectively, for the models with 6,000 and more than 10,000 elements, with 30 vertices included in the touched area.
Another area of research is represented by exploration of linear viscoelastic models and their employment in haptic rendering and interaction. A robust adaptive method for simulating dynamic deformations of a viscoelastic object in real time is described in Debunne et al. (2001). The explicit FEM is assembled and solved for each element independently through a local approximation. The method further employs the adaptive refinement in order to increase the quality of modeling in the vicinity of the interaction.
An algorithm for real-time haptic interaction with linear viscoelastic model is presented in Sedef et al. (2006). The viscoelastic materials are regarded as *materials with memory* as the time-dependent response history is recorded that is evident from the constitutive relation between the stress and strain. Using relations from the linear elasticity and generalized Maxwell solid, the model of linear viscoelasticity is derived resulting in linear time-dependent equation

$$\mathbf{f}_{int}^{n+1} = \mathbf{K}\mathbf{u}^{n+1} + \mathbf{h}_H^{n+1} - \mathbf{K}_H\mathbf{u}^n \quad (13)$$

where $\mathbf{u}^{n+1}$ and $\mathbf{u}^n$ are the vectors of nodal displacement in current and previous time steps, $\mathbf{u}$ and $\mathbf{u}_H$ are the constant stiffness matrix and history stiffness matrix and finally, $\mathbf{h}_H^{n+1}$ is the history vector at the current step computed from the recursive formula for the internal stress variables. The algorithm is divided into two parts: first, the stiffness matrix $\mathbf{K}$ is assembled and inverted and history stiffness matrix $\mathbf{K}_H$ is calculated. Second, the time step loop calculations are executed where in each step $n+1$, the actual history vector $\mathbf{u}_H^{n+1}$ is calculated together with the actual force response $\mathbf{f}^{n+1}$ and displacement field $\mathbf{u}^{n+1}$. Despite the precomputation of the matrix inverse $\mathbf{K}^{-1}$, the calculations proposed for a single time step are still too expensive to be performed in real-time at haptic refresh rate. Therefore, another precomputation is proposed exploiting the linearity and superposition principle. For each surface node, two sets of data are precomputed: first, force response of the node and displacement response of the neighbouring nodes corresponding to unit displacement applied for 30 second to given surfaces node. Second, a unit step force is applied to each surface node for 10 ms

and the recovery displacement response is recorded for the node and its neighbours lasting 30 seconds.

Above, several techniques used for haptic interaction with linear deformation models have been briefly described. It can be concluded that linearity of the mathematical model can be explored very efficiently to achieve the real-time performance as proposed in the papers referenced above. However, as stated before, the linear models provides only limited accuracy and are not generally suitable for large-deformation physical modelling. Therefore, the techniques employing non-linear models proposed recently are briefly introduced in the next section.

2.4 Non-linear models

The shortcomings of linear models are studied in (Picinbono et al., 2001) showing that it is not invariant w. r. t. the rotations and therefore, large deformations cannot be modelled realistically. Therefore, the **non-linear mass-tensor** model is proposed as an extension of the linear version described in the previous section. Geometrical non-linearity is introduced by complete Green-StVenant strain tensor establishing second order relation between strain and displacement. The relation between the stress and strain is, however, linear so material non-linearity is not considered. In order to improve the behaviour of the tissue, anisotropic term is added to the definition of the elastic energy. The formula determining the force applied in a vertex i of a tetrahedron $\mathbf{T}$ is composed of sum of linear, quadratic and cubic term w. r. t. the displacement:

$$\mathbf{f}_p^{\mathbf{T}_i} = 2\sum_j \mathcal{B}_{pj}^{\mathbf{T}_i}\mathbf{u}_j + \sum_{j,k}\left[2(\mathbf{u}_k \otimes \mathbf{u}_j)\mathcal{C}_{jkp}^{\mathbf{T}_i} + (\mathbf{u}_j\mathbf{u}_k)\mathcal{C}_{pjk}^{\mathbf{T}_i}\right] + 4\sum_{j,k,l}\mathcal{D}_{jklp}^{\mathbf{T}_i}\mathbf{u}_l\mathbf{u}_k^\top\mathbf{u}_j \tag{14}$$

The linear term is treated as shown in the previous section: for each tetrahedron, local 3×3 *stiffness tensors* $\mathcal{B}_{pj}^{\mathbf{T}_i}$ are computed for vertices and edges and the results are accumulated to the global stiffness tensors. It is shown that the higher-order terms can be treated similarly: *stiffness vectors* $\mathcal{C}_{jkp}^{\mathbf{T}_i}$ are computed for vertices, edges and triangles in case of the quadratic term whereas *stiffness scalars* $\mathcal{D}_{jklp}^{\mathbf{T}_i}$ are calculated for vertices, edges, triangles and tetrahedra. The structures are then used during the simulation to compute forces for each vertex, edge, triangle and tetrahedron. The Newtonian differential equation

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}(\mathbf{u}) = \mathbf{f} \tag{15}$$

is used as follows: the mass matrix $\mathbf{M}$ and damping matrix $\mathbf{D}$ are replaced by diagonal matrices m_i and γ_i and $\mathbf{f}$ is computed using the Eq. 14. Finally, the equation is decoupled for each vertex separately resulting in

$$m_i\ddot{\mathbf{p}}_i = \gamma_i\dot{\mathbf{p}}_i + \mathbf{f}_i \tag{16}$$

where $\mathbf{p}_i$ is the actual position of a vertex. The explicit integration scheme is used to update the vertex position $\mathbf{p}^{t+1}$ in the following step:

$$\left(\frac{m_i}{\Delta t^2} - \frac{\gamma_i}{2\Delta t}\right)\mathbf{p}_i^{t+1} = \mathbf{f}_i + \frac{2m_i}{\Delta t^2}\mathbf{p}_i^t - \left(\frac{m_i}{\Delta t^2} + \frac{\gamma_i}{2\Delta t}\right)\mathbf{p}_i^{t-1} \tag{17}$$

that can be computed in the time step providing the force $\mathbf{f}_i$ is computed from the stiffness data according to 14.

The modifications of the mesh topology caused by cutting or tearing are implemented by removing the involved tetrahedra and updating the stiffness data of adjacent elements. In order

to increase the performance, a **non-linear model with threshold** is proposed: the model is treated as non-linear only if the current deformation exceeds a defined threshold. Otherwise, linear elasticity is preserved requiring much less computations of the stiffness data. The refresh rate is then increased to 20 Hz from 8 Hz for a model with 6342 tetrahedra Picinbono et al. (2003).

The geometrically non-linear dynamic model of soft tissue with physically linear isotropic material is studied in (Zhuang & Canny, 1999). The finite element method is used to discretize the continuous elasticity problem resulting in time-dependent non-linear Eq. 15 with both $\mathbf{M}$ and $\mathbf{D}$ constant mass and damping matrices and $\mathbf{K}(\mathbf{u})$ stiffness matrix which only depends on the actual displacement vector $\mathbf{u}$. Newmark explicit integration method is used to reformulate the nonlinear Eq. 15 to obtain three linear equations

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \dot{\mathbf{u}}_n \Delta t_n + \frac{1}{2}\ddot{\mathbf{u}}_n \Delta t_n^2 \tag{18}$$

$$(\mathbf{M} + \frac{1}{2}\Delta t_n \mathbf{D})\ddot{\mathbf{u}}_{n+1} = \mathbf{f}_{n+1} - \mathbf{K}(\mathbf{u}_{n+1}) - \mathbf{D}(\dot{\mathbf{u}}_n + \frac{1}{2}\ddot{\mathbf{u}}_n \Delta t_n) \tag{19}$$

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + \frac{1}{2}(\ddot{\mathbf{u}}_n + \ddot{\mathbf{u}}_{n+1})\Delta t_n. \tag{20}$$

The computational bottleneck disallowing the real-time computations is given by the Eq. 19 where inverse of the matrix $(\mathbf{M} + \frac{1}{2}\Delta t_n \mathbf{D})$ is needed. Two variants are proposed in (Zhuang & Canny, 2000): first, mass-lumping approximation is used resulting in diagonalization of both mass matrix $\mathbf{M}$ and damping matrix $\mathbf{D}$. Although the approximate version of matrix $\mathbf{M}$ is acceptable, diagonalization of $\mathbf{D}$ results in loss of viscous property of the material. Therefore, another approach is proposed based on precomputation exploring the fact that both $\mathbf{M}$ and $\mathbf{D}$ are constant and Δt_n is the only varying term depending on the stability of the explicit integration method. The main idea is to restrict the time step Δt_n to small set of values $\{\Delta t_i\}$ and to precompute $(\mathbf{M} + \frac{1}{2}\Delta t_i \mathbf{D})^{-1}$ for each Δt_i. Further optimizations are suggested to minimize the number of floating-point operations needed for the calculations of the equations derived by the Newmark method. The total length of 0.06 s needed for a single time step is reported in (Zhuang, 2000) for a uniform mesh with 1331 elements. The haptic refresh rate is achieved by the force interpolation using two adjacent simulation steps.

Both geometrically and physically non-linear model was presented in (Wu et al., 2001). The finite element formulation is given for the Mooney-Rivlin and Neo-Hookean materials using the complete non-linear strain tensor. Dynamic scenario is considered based on the Eq. 15 where both mass and damping matrices are diagonalized. The explicit integration scheme is used in the simulation. The main contribution is represented by implementation of on-line mesh refinement using **dynamic progressive meshes**. To achieve the real-time refresh rate, all the invariant components of the model, such as constant matrices $\mathbf{M}$ and $\mathbf{D}$, Jacobian matrix $\mathbf{J}$ needed for the calculation of the stress, are precomputed off-line together with their inverses. Moreover, the constant data structures needed for the mesh refinement and coarsening are identified and precalculated. For a model of tissue with 2200 vertices, refresh rate of 20 frames per second is reported using Pentium III PC running at 800 MHz. The stability of the explicit integration is improved by **multigrid method** proposed in (Wu & Tendick, 2004). A multigrid time integrator is developed to perform the integration of the Eq. 15 using *restriction*, *interpolation* and *solution* operator to implement the projections between multiple unstructured independent meshes. Finally, the parallel version of the multigrid-based simulation is described in (Wu et al., 2004). The performance of the solver is evaluated together

with the stability of the simulation showing promising results suitable for the real-time processing.
A new approach based on precomputation is proposed (Barbič & James, 2005) based on **formal model reduction** of StVenant-Kirchhoff. The reduction, which is suitable for large deformations but only a small local strain, is based on projection of the displacement vector $\mathbf{u}$ with $3n$ elements onto vector $\mathbf{q}$ with r elements where $r \ll 3n$ using time-independent basis represented by a $3n \times r$ matrix $\mathcal{P}$: $\mathbf{q} = \mathcal{P}\mathbf{u}$. The nontrivial task of generation of the basis $\mathcal{P}$ is studied in detail and method based on combination of *linear vibration modes* and mass-scaled PCA is suggested. The reduced equation of motion is derived resulting in

$$\ddot{\mathbf{q}} + \tilde{\mathbf{D}}(\mathbf{q}, \dot{\mathbf{q}}) + \tilde{\mathbf{R}} = \tilde{\mathbf{f}} \tag{21}$$

where $\tilde{\mathbf{D}}$ and $\tilde{\mathbf{R}}$ are r-dimensional vectors representing the reduced forces. Further, it is shown that for the StVenant material, the components of the reduced internal forces $\tilde{\mathbf{K}}$ are cubic polynomials in $\mathbf{q}$:

$$\tilde{\mathbf{R}} = \mathbf{P}^i \mathbf{q}_i + \mathbf{Q}^{ij} \mathbf{q}_i \mathbf{q}_j + \mathbf{S}^{ijk} \mathbf{q}_i \mathbf{q}_j \mathbf{q}_k \tag{22}$$

where $\mathbf{P}^i$, $\mathbf{Q}^{ij}$ and $\mathbf{S}^{ijk}$ are constant polynomial coefficients that can be efficiently precomputed; e. g. for object with 14,444 elements, the time of 97.4 minutes needed for the precomputations on single 3 GHz Pentium workstation is reported. The polynomial coefficient can be used also to compute the derivative of $\tilde{\mathbf{R}}$ w. r. t. $\mathbf{q}$ corresponding to the reduced tangent stiffness matrix. During the simulation, the reduced Eq. 21 is integrated in time using implicit integration Newmark method. In each time step, only single Newton-Raphson is executed instead of iterating until convergence is achieved. The Newton-Raphson method requires calculation of reduced stiffness and tangent stiffness matrix that is done using the precomputed coefficients $\mathbf{P}^i$, $\mathbf{Q}^{ij}$ and $\mathbf{S}^{ijk}$ and dense linearized $r \times r$ system is solved. The model reduction is applied to objects having between 2000 and 14500 element resulting in systems with $r \in \{12, 15, 30\}$. The refresh rate ranging between 45 Hz to 470 Hz is achieved using GPU acceleration. The approach based in the modal reduction is reused in (Barbič & James, 2008) where contact modeling is analyzed for 6DoF haptic rendering of object with complex geometry.
The approach published in (De et al., 2006) is based on non-linear elasticity and mesh-free discretization known as **point-associated finite field**. Instead of FE discretization, the PAFF-based discretization of the domain provided by scattered points is used together with interpolation generated by least-squares method. Comparing to standard FEM, PAFF introduces additional error in the simulation. Moreover, the non-linearity of the tissue in the vicinity of the operation tool is considered, whereas linear model is employed on the rest of the body. Likewise in FEM, the PAFF discretization leads to a system of non-linear algebraic equation that can be solved by Newton-Raphson methods where in k-the iteration, the correction of the displacement vector $\mathbf{\Delta u}^{(k)}$ is computed by solving a linearized system and the displacement vector is updated, i. e.

$$\mathbf{K}_T(\mathbf{u}^{(k-1)})\mathbf{\Delta u}^{(k)} = \mathbf{f} - \mathbf{K}(\mathbf{u}^{(k-1)}) \tag{23}$$

$$\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} + \mathbf{\Delta u}^{(k)} \tag{24}$$

where the tangent stiffness matrix $\mathbf{K}_T$ dependent on $\mathbf{u}$ is composed of the constant stiffness matrix $\mathbf{K}_0$ from the linear small deformation problem and varying component $\mathbf{K}_S(\mathbf{u})$. Clearly, the iterative calculations composed of assembling and inverting the large matrix $\mathbf{K}_T$ cannot be done in real-time and therefore it is assumed that there is a scalar s such that $\mathbf{K}_T = s\mathbf{K}_0 + \mathbf{\Delta K}_S$

and s can be found easily minimizing the Frobenius norm $\|\Delta\mathbf{K}_S\|_F$. Then, the time-consuming solution given by Eq. 23 can be replaced by computation $\Delta\mathbf{u}^{(k)} = \frac{1}{2}\mathbf{K}_0^{-1}(\mathbf{f} - \mathbf{K}(\mathbf{u}^{(k-1)})$. Since the matrix $\mathbf{K}_0$ is constant, its inverse can be precomputed before the iteration starts. The total length of computations needed to calculate the scalar s is about 0.14 s on Pentium 4 running at 2 GHz, provided the inverse $\mathbf{K}_0^{-1}$ has been already precomputed taking about 20 minutes.
Another approach based on standard finite element method is presented in (Deo & De, 2009). First, commercial software is used to compute a set of deformations corresponding to a selected boundary-value problems. The precomputed data are then used for training of *radial basis function networks*, i. e. neural networks being associated to each node of the FE mesh. The neural network is then used to determine the reaction force for given node displaced by the surgical tool. The error of the approximation given by a network composed of 50 neurons after training on 3000 precomputed sample is given being 5% when processing 5000 samples not included in the training set. The real-time computation of the force response takes about 1.5 μs for a 3D model having 2839 elements after 19 hours of precomputations.
A novel algorithm for non-linear real-time FE modelling is presented in (Taylor et al., 2007). It is based on **total Lagrangian explicit dynamics** (TLED) (Miller et al., 2007). The algorithm is composed of two phases: first, all the element shape function derivatives are precomputed together with the mass matrix. Second, the real-time simulation is executed where in each time step, prescribed displacements are applied to the model and total nodal forces are precomputed together with strain-displacement matrix and stresses. These are incorporated in the central difference method to compute the update displacement vector. The algorithm is re-implemented within the SOFA framework (Allard et al., 2007) on nVidia GPU accelerators using CUDA programming language as shown in (Comas et al., 2008). The performance of the CUDA-based implementation is demonstrated on medical application — cataract surgery. Employing the FE model simulating non-linear, viscoelastic and anisotropic behaviour of the lens, undergoing large deforamations, one step of the TLED algorithm is reported to run at 284 Hz and 87 Hz for two models of lens having 4058 and 47633 elements, respectively.

In this section, the algorithms employing non-linear models for modelling the deformations in real-time have been briefly presented. Also in this case, several techniques rely heavily on the precomputations, so the time-consuming computations needed for realistic non-linear modeling are performed separately from the high-frequency haptic loop. However, it turns out that the techniques are often restricted to a certain type of a material (e. g. for which the model reduction can be performed) or they are limited by the instability in the case when explicit time-integration schemes are employed.

3. Haptic Interaction Based on Precomputation of Configuration Spaces

3.1 Approximation from Discretized Spaces

Similarly as in the case of the algorithms surveyed in the section 2, our algorithm is focused on the solution of the problem how to combine the high refresh rate of the haptic interaction and computationally expensive finite element calculations which are needed to computed the deformations and response forces. There are two main goals in the design of the algorithm:

1. the deformation model is based on non-linear static finite element formulation and there are no *a priori* assumptions about the complexity of the mathematical model, however, the topological changes are not considered;

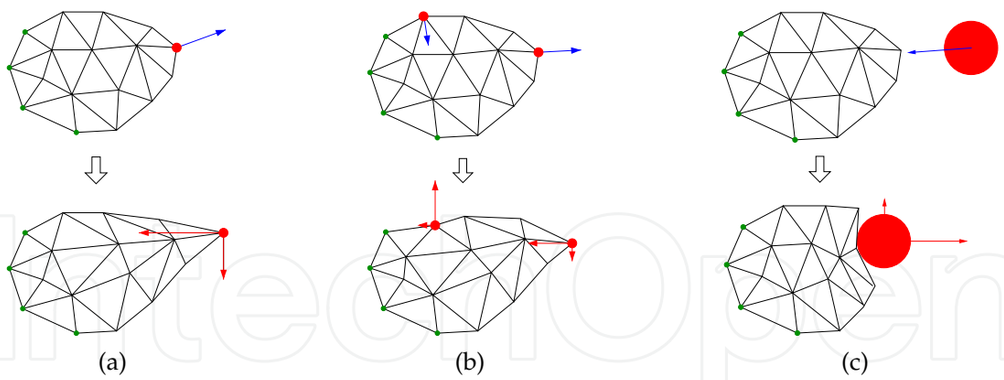

Fig. 2. Interaction modes: (a) point interaction, (b) multi-point interaction, (c) probe interaction

2. the computations needed for rendering the response force can be done within the haptic loop computed on today's computers.

The algorithm is designed for the **displacement-driven interaction**: it is supposed that the positional data $\mathbf{p}$ provided by the haptic device are somehow mapped to a non-homogeneous boundary conditions $\mathbf{b}_N$ which are applied within the boundary-value problem being solved to obtain the response forces and the overall deformation of the object. The positional data $\mathbf{p}$ are therefore regarded as a **control parameter** of the simulation and mathematically, it is represented by a vector which fully determines the position and the orientation of all the haptic devices involved in the simulation. The mapping between the positional data $\mathbf{p}$ and the non-homogeneous boundary conditions $\mathbf{b}_N$ depends on the type of the interaction between the haptic device and deformable body. We distinguish two basic types of the interaction depending on this coupling as follows.

Single-point interaction: the haptic device is represented by a single point in 3D space called *haptic interaction point* (HIP), so the positional data $\mathbf{p}$ are given by the spatial coordinates of the point. If the point collides with the deformable object, it snaps to a vertex of the FE mesh representing the body. Without loss of generality, it is assumed that HIP snaps to the closest vertex of the mesh which becomes **active node**. The mapping between the positional data $\mathbf{p}$ and the non-homogeneous boundary conditions $\mathbf{b}_N$ is straightforward: the displacement of the active node is given directly by the actual position of the haptic device. The scenario is depicted by Fig. 2(a).

An extension of the approach can be proposed: e. g. **multiple-point interaction** is considered if there are at least two HIPs (manipulated by e. g. two haptic devices) each being attached to one active node. In this case, the control parameter $\mathbf{p}$ contains the prescribed positions of all the active nodes which are attached to the corresponding HIPs and the mapping to the boundary conditions is again straightforward. The scenario is illustrated by Fig. 2(b).

Probe interaction: the HIP is represented by a **probe** which moves in the space. If the probe collides with the FE mesh, the nodes in collision are identified and the boundary conditions are assembled. Mathematically, the positional data $\mathbf{p}$ are given by the position

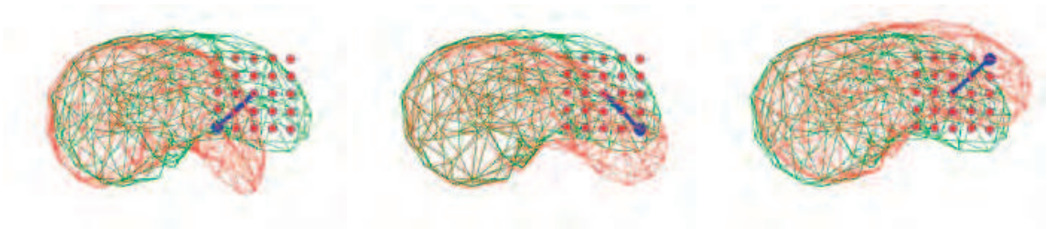

Fig. 3. Three configurations of the deformable body stored in the grid. Two-dimensional projection is used for the simplicity.

of the probe. However, the mapping between $\mathbf{p}$ and non-homogeneous boundary conditions $\mathbf{b}_N$ is not straightforward anymore, as the displacements of the nodes being in collision with the probe are not a priori known. To obtain the boundary conditions, some advanced methods developed in the contact modelling can be applied, e. g. using the **gap function** as studied in Saupin et al. (2008). The scenario with spherical probe being manipulated by a 3DoF haptic device sourcing its position in 3D space is depicted by Fig. 2(c). Alternatively, the probe with more complicated or irregular shape can be employed, e. g. representing a surgical tool. In this case, not only the position but also the orientation of the probe can be given by the positional data $\mathbf{p}$.

Let us now focus on the single-point interaction, so the vector $\mathbf{p}$ represents directly the prescribed positions for some given set of nodes. Therefore, given a FE mesh discretizing the body being fixed in the space, the finite element method is used to compute a generally nonlinear mapping $\mathbf{p} \rightarrow (\mathbf{h}, \mathbf{u})$ where $\mathbf{h}$ and $\mathbf{u}$ are the response forces and nodal displacements corresponding to the prescribed positions $\mathbf{p}$. It is supposed that the mapping $\mathbf{p} \rightarrow (\mathbf{h}, \mathbf{u})$ is unique, i. e. for given prescribed positions $\mathbf{p}$, the boundary-value problem has a unique solution. This is given by the ellipticity of the problem being considered. Formally, the state of the model in a given step of the simulation is fully given by a configuration C which is defined as a triple $\langle \mathbf{p}, \mathbf{h}, \mathbf{u} \rangle$.

If the vector of the prescribed positions is updated, i. e. $\mathbf{p} \rightarrow \mathbf{p}'$ due to the change in the position of HIP, the corresponding response forces $\mathbf{h}'$ and the deformation $\mathbf{u}'$ of the body are computed, i. e. the *transition* $C \rightarrow C'$ between the two configurations is constructed. 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. Finally, the set of all configurations with defined mapping $C \rightarrow C'$ is denoted as a configuration space $\mathcal{C}$. Apparently, $\mathcal{C}$ is infinite and continuous, however, it is *bounded*: i. e. only configurations having the amplitude of reacting forces $|\mathbf{h}|$ lower than some finite constant F_{max} are included in the space. This is given by the physical limitation of each haptic device, as the force which can be displayed by the device is bounded by some F_{max} which ranges from 10 N to 20 N for today impedance-control devices.

In terms of the configuration spaces introduced above, the main idea of our approach is to construct a discretization — a finite subspace $\mathcal{D} \subset \mathcal{C}$, so that $\mathcal{D}$ can be constructed in finite time and stored. Considering the single-point interaction, it is assumed that the discretization is provided by a regular uniform grid $\mathcal{G}$ where each point $\mathbf{g} \in \mathcal{G}$ is given by three spatial coordinates (g_x, g_y, g_z). The center of the grid is denoted as $\mathbf{g}_0$ and there are two parameters determining the distributions of the grid points: $r_{\mathcal{G}}$ is the radius of the grid, i. e. the distance between the grid center and the outermost grid point, and $d_{\mathcal{G}}$ is the distance between two

Fig. 4. Two deformations of the object for the identical position of the probe (depicted by the filled red circle) showing the non-flatness of the probe interaction model. On the left, the probe comes from the space below the deformable body, on the right it comes from the space above the body (the directions are depicted by the blue arrows.

adjacent grid points being measured along the coordinate axes. It is important not to confuse the regular uniform grid $\mathcal{G}$ with the unstructured finite element mesh of the deformable body. Provided the active point has been already selected, the grid $\mathcal{G}$ is employed by the precomputation-approximation scheme which composed of two separated phases as follows:

Off-line phase. The discrete configuration space $\mathcal{D}_\mathcal{G}$ is constructed using $\mathcal{G}$: the active node is placed to each grid point $\mathbf{g}$ and the corresponding configuration $C_\mathbf{g} = \langle \mathbf{g}, \mathbf{u}, \mathbf{h} \rangle$ is constructed and inserted into $\mathcal{D}_\mathcal{G}$. In other words, it is simulated that the control parameter is set to each $\mathbf{g}$ and the corresponding response force and deformation are computed. The grid of configurations for the single-point interaction is demonstrated by Fig. 3.

The points of the grid are traversed systematically: having a configuration $C_\mathbf{g}$ stored in the point $\mathbf{g}$, the transitions $C_\mathbf{g} \to C_{\mathbf{g}'}$ for all adjacent grid points $\mathbf{g}'$ are constructed and this process is recursively repeated until there is a configuration associated to each grid point of the grid $\mathcal{G}$. The algorithm of systematic grid traversal together with distributed version is described in section 3.2.

On-line phase. The haptic interaction is executed for given active node, approximating the response forces and overall deformation for arbitrary position of the haptic device using the precomputed data. More precisely, in each step of the haptic loop, the positional data $\mathbf{p}$ are acquired and the corresponding configuration $\tilde{C}_\mathbf{p} = \langle \mathbf{p}, \mathbf{u}, \mathbf{h} \rangle$ is computed by interpolation from the precomputed configurations stored in $\mathcal{D}_\mathcal{G}$. The interpolation techniques suitable for the real-time interaction are studied in section 3.3.

The concept of the space-division scheme presented above can be used also for other types of the interaction. First, in case of the multi-point interaction, the higher-order grid must be employed; e. g. for two active points A and B, the six-dimensional grid is considered, where each point is given as six-tuple $(g_x^A, g_y^A, g_z^A, g_x^B, g_y^B, g_z^B)$. During the off-line phase, the configurations are computed for all the grid points, i. e. for all the positions of the two points A and B within the positions of the discretized space. In the on-line phase, the response forces are interpolated for each active node separately using the precomputed data.

Second, in case of the probe-interaction, the adaptation is a bit more complex. Let us assume a spherical probe, as shown in the Fig. 3. The position of the probe is given by a three-dimensional vector containing the position of the probe center. Therefore, the space containing the deformable object is discretized by three-dimensional grid in the same way as in the single-point interaction. However, it is important to realise that for one position of the probe, there can be multiple distinct configurations as shown in Fig. 4; the overall deformation and reaction forces depend on the direction $\mathbf{d}$ of the probe motion. Therefore, **enriched regular grid** $\mathcal{G}^*$ is introduced: it is based on the grid $\mathcal{G}$, but additionally, for given grid point $\mathbf{g}$, there

are multiple levels $l_1, l_2, \dots l_L$ corresponding uniquely to particular directions. During the off-line phase, the enriched grid is traversed systematically: being in a configuration $C^l_{\mathbf{g}}$ stored in a grid point $\mathbf{g}$ and a level l, the transition towards a grid point $\mathbf{g}' = \mathbf{g} + \mathbf{d}$ is constructed as follows:

1. the configuration C' corresponding to the position $\mathbf{g}'$ is computed taking into account the original position $\mathbf{g}$ from which the probe has arrived.
2. The configuration C' is stored as $C^{l'}_{\mathbf{g}'}$, where l' is the identifier of the level associated with the direction $\mathbf{d} = \mathbf{g}' - \mathbf{g}$.

Having the configurations stored in the enriched grid $\mathcal{G}^*$, the space can be traversed easily during the on-line phase, since the actual direction can be used to identify the level where the target configuration is stored.
Clearly, the concept of levels can be combined with higher-order grids for the case when there are more positional data identifying the state of the probe. For example if rotations are given for the probe using for example 6DoF device, the 6D enriched grid can be established, where the grid points corresponds to the positional and rotational data and the levels are used to handle the transitions between various positions and orientations of the probe.
In the following section, the algorithms for both the construction of the configuration space and interpolation from the precomputed data are briefly described. For the sake of simplification, the algorithms for single-point interaction are presented. The reader interested in concept of enriched grid used for the real-time haptic interaction with precomputations can find detailed description in Křenek (2003); Peterlík (2009).

3.2 Algorithm for the Space Construction

The generation of the configuration space is a time-consuming procedure, since the configurations for the entire grid must be systematically constructed. The procedure is studied on two levels: first, it is necessary to specify the method of calculation of single configuration C. Second, the algorithm for the overall construction of the configuration space must be given, i. e. the order in which the computations of the transitions are done to generate the entire space.
The computation of one transition $C_A \rightarrow C_B$ is performed as follows. According to the assumption formulated in the previous section, the non-linear finite element formulation is employed. The simulation is driven by the control parameter $\mathbf{p}$ which represents the vector of the prescribed positions for a given set of nodes. Provided, the configuration C_A is known as $C_A = \langle \mathbf{p}_A, \mathbf{h}_A, \mathbf{u}_A \rangle$ whereas the new position for the configuration C_B is $\mathbf{p}_B$, the vector of the prescribed displacements in C_A and C_B can be computed easily as $\mathbf{u}^*_A = \mathbf{p}_A - \mathbf{p}_0$ and $\mathbf{u}^*_B = \mathbf{p}_B - \mathbf{p}_0$ where $\mathbf{p}_0$ is the rest position of the active nodes. Within the finite element formulations, the vector $\mathbf{u}^*$ represents a non-homogeneous Dirichlet condition. There are several methods to include the condition into the system — elimination, penalization or Lagrange multipliers. In our case, the method of the Lagrange multipliers is employed, since it directly yields the response forces which are to be determined. The method results in the augmented non-linear system of algebraic equations $\hat{\mathbf{K}}([\mathbf{u}|\boldsymbol{\lambda}]) = [\mathbf{f}|\mathbf{u}^*]$ which must be solved iteratively. In our approach, a technique combining the incremental loading and Newton-Raphson methods is considered. First, using $\mathbf{u}^*_A$ and $\mathbf{u}^*_B$, an **incremental loading** path $\{\mathbf{u}^*_A = \mathbf{u}^*_{(0)}, \mathbf{u}^*_{(1)}, \dots \mathbf{u}^*_{(M)} = \mathbf{u}^*_B\}$ is constructed and the sequence of non-linear systems $\hat{\mathbf{K}}([\mathbf{u}|\boldsymbol{\lambda}]) = [\mathbf{f}|\mathbf{u}^*_{(i)}]$ with initial estimation $\mathbf{u}^*_{(i-1)}$ is solved for $i = 1, \dots, M$.
Further, the solution in each step is obtained by **Newton-Raphson method**: having some estimation of both the nodal displacement vector $\mathbf{u}^{(j)}$ and Lagrange multipliers $\boldsymbol{\lambda}^{(j)}$ in the j-th

iteration of the method, new estimations are computed as

$$\mathbf{u}^{(j+1)} = \mathbf{u}^{(j)} + \Delta\mathbf{u}^{(j+1)} \quad (25)$$

$$\boldsymbol{\lambda}^{(j+1)} = \boldsymbol{\lambda}^{(j)} + \Delta\boldsymbol{\lambda}^{(j+1)} \quad (26)$$

where both corrections $\Delta\mathbf{u}^{(j+1)}$ and $\Delta\boldsymbol{\lambda}^{(j+1)}$ are computed by solving the linearized system

$$\hat{\mathbf{K}}'([\mathbf{u}^{(j)}|\boldsymbol{\lambda}^{(j)}]) \begin{bmatrix} \Delta\mathbf{u}^{(j+1)} \\ \Delta\boldsymbol{\lambda}^{(j+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{u}^{*}_{(i)} \end{bmatrix} - \hat{\mathbf{K}}([\mathbf{u}^{(j)}|\boldsymbol{\lambda}^{(j)}]) \quad (27)$$

where $\mathbf{u}^{*}_{(i)}$ is the vector of the prescribed displacement given in the actual step i of incremental loading and $\hat{\mathbf{K}}'([\mathbf{u}^{(j)}|\boldsymbol{\lambda}^{(j)}])$ is the tangent stiffness matrix computed as the spatial derivative of the non-linear mapping $\hat{\mathbf{K}}([\mathbf{u}|\boldsymbol{\lambda}])$. The estimation for the first step of the Newton-Raphson method in the first step of the incremental loading method is taken from the starting configuration C_A: $\mathbf{u}^{(0)}_{(0)} = \mathbf{u}_A$ and $\boldsymbol{\lambda}^{(0)}_{(0)} = \mathbf{h}_A$. There are two criteria for the Newton-Raphson method to stop: first, if the residual given by the right-hand side of the Eq. 27 is less then a chosen ε in some iteration of the Newton-Raphson method, the resulting solution converges and is considered to be a valid data. Second, the iteration process is also stopped after N maximum iterations also if the condition above does not hold. In this case, the calculation is considered as non-convergent and the data is considered to be invalid. After obtaining the results in the last step M of the incremental loading path, new configuration C_B is set as $C_B = \langle \mathbf{p}_B, \boldsymbol{\lambda}_{(M)}, \mathbf{u}_{(M)} \rangle$. It is to be noticed that besides the final configuration C_B the intermediate configurations $C_{(1)}, \ldots C_{(M-1)}$ are computed due to the incremental loading. These can be stored in an additional file, because they can be utilized by a special type of the interpolation as shown in the following section.

On the higher lever, the order in which the transitions are computed is to be specified. In Peterlík (2008), the *layer-based technique* is proposed for the efficient construction of the configuration spaces. Originally, the technique was designed for sequential processing, however, it could be also executed in distributed environment, as the independent transitions within one layer can be computed concurrently. Further, in Filipovič et al. (2009) a distributed algorithm is proposed for state space generation based on client-server scheme. Briefly, the client is represented by a *scheduler process*, whereas on the server side, there is a pool of *solver processes* which compute the transitions simultaneously. All the processes involved in the computations communicate among themselves using message-passing.

The main role of the scheduler is as follows. It maintains two data-structures:

1. table t_C of all configurations which are to be computed. In the table, each configuration is marked either as *unknown*, *assigned* or *computed*;
2. table t_S of all the solver processes being in the pool. Each server is marked either as *idle* or *busy*.

Initially, each configuration is marked as unknown and each solver is marked as idle. In the first step, the scheduler finds such a position $\mathbf{g}$ within the grid $\mathcal{G}$ that there is no collision between the HIP and the FE mesh, i. e. the both the force $\mathbf{h}$ and nodal displacement $\mathbf{u}$ are zero and therefore, the corresponding configuration is marked as known. Then, the algorithm executes as follows:

1. If there is an idle solver S_i, choose an unknown configuration C' such that the transition $C \rightarrow C'$ can be constructed, i. e. the configuration C is marked as computed. If there is such C', send the configuration C to the solver S_i together with the target prescribed displacement, mark C' as assigned and S_i as busy.
2. If there is still some unknown configuration, but no transition can be actually computed, wait until some of the processes completes the computations and sends the notification message. Let's assume, that a solver S_i sends a notification about completing the computation of a configuration C'. In this case, mark S_i as idle and mark configuration C' as computed.
3. Repeat the steps 1. and 2. until all configurations are marked as computed.

On the other side, the protocol of the solver process is simpler: the computation of a transition $C \rightarrow C'$ is initialized by a message from the scheduler, determining the initial configuration C and the target prescribed displacement. After the computation is completed, the configuration is stored and the notification message is sent to the server. Anytime the configuration is needed by some solver, the scheduler can provide the identifier of the solver which keeps the configuration. The concept of notifications is again utilized to obtain the configuration from some solver in order to start the computation of a new transition.
The construction of the configuration space is completed, providing all the configurations are marked as computed. This also implies that all the solvers are also marked as idle and all the configurations have been stored on a disk and can be used for the haptic interpolation being executed in the second phase.

3.3 Real-time Interaction

Basically, the real-time interaction is implemented by a single process called *force interpolator*, which first reads all the precomputed data from the disk and then, for the actual position of the HIP, it identifies the adjacent points of the precomputed configuration space and computes the interpolation to approximate the actual force. The interpolation can be performed inside the haptic loop, as it requires only a small number of operations with data stored in the computer memory.
It is important to note, that the precomputed data are utilized also for the approximation of the object deformation for the visualization purposes. The interpolation of the surface nodes is computed by visual interpolator and rendered by visual thread running at 25 Hz. Albeit the interpolation of larger data set must be performed, the refresh rate of the visual rendering runs on much lower frequency.
In this part we focus on interpolation of the response force during interaction. Without loss of generality, it is assumed that the force is represented by three components Each component (f_x, f_y, f_z) is interpolated from the precomputed forces separately. In the following, two different methods are briefly studied: *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 $C_{\mathbf{g}}$ 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 $\mathbf{g}_i$ of the 3D grid must be first determined. For the simplest case of *trilinear interpolation*, eight neighboring nodes are needed for the interpolation. The main advantage of the tri-linear interpolation is that the computations are very simple, since only a small number of operations is needed. Precisely, the interpolation of a single force component requires about 250 floating point operations, regardless the size of the grid. Therefore, the interpolation of the force vector can be performed

on single-core computer easily. On the other hand, there are some drawbacks of this method. Although the interpolation is continuous, it is not smooth in the moment of switching between cells: the actual values are instantly interpolated from different values. E.g. when traversing the cell boundary via the face, four vertices are switched, whereas when traversing via corner, even seven vertices must be switched resulting in non-smoothness of the interpolation in the moment of cell switching.
Another alternative within the frame of the polynomial method is *tricubic interpolation*. In this case, 64 precomputed configurations are needed for approximating a single value. 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. Comparing to the tri-linear interpolation, the complexity of cubic interpolation increased, as it requires about 2.000 floating-point operations to be evaluated for the interpolation of one component of the force vector. Nevertheless, this number is again independent on the size and density of the grid. When comparing to the performance of today's processors, the interpolation of the force vector can be easily computed within the haptic loop running on high frequency. The problem with non-smooth behaviour is not solved completely, nonetheless, the smoothness is significantly improved, as in the worst case (HIP crossing the grid point) 27 out of 64 interpolation points remain unchanged. In the best case, when the face is traversed, even 48 out of 64 interpolation points remain unchanged.

Radial-basis function interpolation. There are two reasons why a method interpolating from scattered (irregularly distributed) data is desired:

- both the configurations in the points of the regular grid and in the intermediate states can be utilized and
- before the interpolation takes place, the configurations computed in non-convergent iterations can be excluded from the precomputed data.

To meet these requirements, *radial-basis function* (RBF) 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:

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} w_i \phi(|\mathbf{x} - \mathbf{x}_i|) \tag{28}$$

where ϕ is a chosen function, e. g. $\phi(r) = r$ or $\phi(r) = r^3$ determining linear and cubic interpolation, respectively.
The weights w_i can be computed during the precomputation phase by solving and decomposing a linear system which is assembled by substituting the known values $f(\mathbf{x}_j)$ into Eq.28. Having the weights precomputed, the interpolation for the actual position can be computed quickly even for large number of interpolation points. Unlike the polynomial interpolation, the time complexity of the computations inside the haptic loop linearly depends on the number of nodes in the grid. Nevertheless, the computations can be performed real-time for sufficiently large grids using today computers. As the 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.

As shown above, polynomial as well as the RBF interpolation can be explored in the on-line phase of the scheme to approximate the actual configuration in real-time using the precomputed data. In the following section, some details concerning the implementation and evaluation of the algorithms presented before are given.

3.4 Evaluation and Discussion

In our prototype implementation, both the scheduler and pool of the solvers were implemented in C++ programming language. The communication between the remote processes was provided by Message-Passing Interface (MPICH implementation ver. 1.2.7 communicating over sockets). The configurations represented by the various deformation of the object were using using GetFEM, an open-source FEM library. The solution of the linearized system computed in each iteration of the Newton-Raphson method was performed by MUMPS linear solver (see P.R.Amestoy et al. (2000)). Further, the force interpolator was implemented for the interpolation techniques presented in section 3.3. The interpolation of the forces was stably running at a frequency of 1,000 Hz on a workstation equipped with 2× Dual Core AMD Opteron 285 processor. Similarly, precomputed nodal displacements were utilized by shape interpolator computing the actual deformation of the body for the visualization purposes running at 25 Hz.

The experiments evaluated in the next part of this section were performed on 3D model of human liver obtained from the INRIA repositories. The model was meshed by TetGEN mesh generation tool resulting in two meshes with 1777 elements (501 nodes) and 10280 elements (2011 nodes), respectively. The real length of the model was about 22 cm. We use both Mooney-Rivlin and StVenant-Kirchhoff material laws; in the case of Mooney-Rivlin, the incompressibility conditions were imposed by mixed formulation.

Extensive testing has been performed to validate the approach based on the precomputation of the configuration spaces. The evaluation can be divided into two parts as follows. First, the accuracy of the methods has been studied. For this purpose, a large number of configurations has been computed and stored for a random values of the positional data. The approximated counterparts have been generated by the interpolation of the precomputed spaces, the forces and displacements have been compared and evaluated. The mean and maximum errors have been calculated using the large set of computed data as shown in Peterlík & Matyska (2008). The tests have been performed for four different densities of the grid and 4 different interpolation methods. It was concluded that the density of the grid is the important factor, nevertheless, it can be compensated by using RBF interpolation which gives good results also for sparse grids. For example, the tri-linear interpolation on the dense grid ($d_{\mathcal{G}} = 6.667\,mm$) results in relative mean error below 1%, which is roughly the same as the results obtained by the RBF cubic interpolation on the sparse grid ($d_{\mathcal{G}} = 20\,mm$). Similar results were obtained also w. r. t. the maximum errors: tri-linear interpolation on the dense grid results in maximum relative error achieving 30%, whereas the RBF interpolation on the coarse grid results in maximum relative error under 20%.

The second part of the testing focused on the precomputation phase. Here, the behaviour of the distributed algorithm was studied w. r. t. the scalability and speed-up. It was shown that the algorithm scales almost linearly for 4, 8, 16, 32 and 64 solver processes in the pool. Furthermore, the experiments with geographically distributed environment were performed using two clusters being located more than 300 km from each other. It was confirmed that the algorithm is resistant to latencies as the scalability was not affected by the distance between the two clusters. Finally, the total length of the computations was studied. The cubic com-

plexity of the computations w. r. t. the resolution of the grid $\mathcal{G}$ was confirmed. Nevertheless, it was shown that also for detailed models, the precomputation can be done in time which is acceptable. For example, using the cluster with 64 CPUs, the construction of the configuration space on grid with 14146 grid points ($d_{\mathcal{G}} = 6.667\,mm$) took less than 3 hours for a model with 10270 elements employing the Mooney-Rivlin material with incompressibility conditions. For the comparison, construction of the space for grid with 514 nodes ($d_{\mathcal{G}} = 20\,mm$) using the same mesh and model took less than 30 minutes.

The quantitative evaluation and detailed discussion of the results obtained for the method presented in this chapter can be find in Peterlík (2009); Peterlík et al. (2010), where also the convergence analyses for various materials, boundary conditions and loading paths are investigated.

So far, the tests have been performed for the single-point interaction, since in that case, only the flat 3D grid is constructed during the off-line phase. It is clear, that other types of the interpolation can be considered, however, at the cost of increased computational complexity: in the case of multiple-point interaction, each degree of freedom yields additional dimension of the grid, whereas the probe interaction introduces additional levels for each grid point. In each case, the number of transitions that must be constructed to traverse the entire configuration space increases rapidly. Therefore, a modification of the approach has been presented in Filipovič et al. (2009). The configuration space is not constructed in advance in a separated phase, however, the new configurations are generated directly during the real-time interaction. The "on-line" version of the space construction assumes the haptic interaction point to be connected to sufficient computational resources such as cluster or grid and it introduces some restrictions concerning the maximum speed of the haptic device during the interaction. On the other side, the time-consuming precomputation phase is not needed anymore and therefore, more complex versions of the grid (additional dimensions and levels) can be considered. A preliminary evaluation of the on-line generation of configuration spaces can be found in Peterlík & Filipovič (2010).

4. Conclusions

In this chapter, we focused on haptic rendering of objects with complex behaviour. The study aimed at deformable bodies which are difficult to model in real-time, provided realistic and physically-based simulation of deformations is desired as in the case of surgical simulators. First, a short overview of the simulation methods was given, emphasizing the computational complexity of the calculations. The two sources of the non-linearity that emerge in the deformation modeling were briefly described and the effect of the linearization was shown. Then, a survey of methods proposed over the last decade was given: it was shown that the precomputation usually plays an important role in design of algorithms combining computationally demanding calculations and real-time response. The key concepts used to overcome the high refresh rate needed for stable haptic rendering were described separately for linear and non-linear models.

In the second part of the chapter, the approach based on the precomputation of the configuration spaces was described. First, the haptic setting was introduced for single-point, multi-point and probe interactions. After introducing the notion of configuration and transition, it was shown that interaction with the deformable objects can be regarded as traveling through configuration spaces. The discretization of such spaces was proposed together with corresponding algorithms for its construction and approximation. The feasibility of the approach

was briefly sketched summarizing the main results of the extensive evaluation. Finally, the on-line version of the algorithm was briefly discussed, showing the direction of further research towards more complex types of interaction between the user and deformable body.
The development in the area of the soft tissues foreshadows that precomputation can still play an important role in the haptic rendering of complex objects. Nevertheless, the algorithms based on direct on-line computations are becoming still more and more attractive, as they allow for flexible modification of the model parameters during the interaction without necessity to recompute the data. The design of such algorithms is also encouraged by the advent of powerful accelerators such as GPGPUs, which significantly increases the performance of single workstation that can be now used for expensive numerical calculations. Therefore, it is possible to conclude that the physically-based deformation modeling in combination with haptic rendering is a promising area where a sharp increase in the quality of simulation can be expected. This will mainly concern the design of visco-elastic materials being in accordance with *in vitro* experiments, heterogeneous models describing the internal structure of the organs, advanced contact modeling considering the interaction between the organs, more precise FE approximations using the meshes composed of large number of special elements, advanced techniques allowing operations such as cutting, tearing or burning the tissue and others.

5. References

Allard, J., Cotin, S., Faure, F., Bensoussan, P.-J., Poyer, F., Duriez, C., Delingette, H. & Grisoni, L. (2007). Sofa an open source framework for medical simulation, *Medicine Meets Virtual Reality (MMVR'15)*, Long Beach, USA.

Barbič, J. & James, D. L. (2005). Real-time subspace integration for st. venant-kirchhoff deformable models, *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, pp. 982–990.

Barbič, J. & James, D. L. (2008). Six-dof haptic rendering of contact between geometrically complex reduced deformable models, *IEEE Trans. Haptics* **1**(1): 39–52.

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

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

Chai, J., Sun, J. & Tang, Z. (2001). Hybrid fem for deformation of soft tissues in surgery simulation, *MIAR '01: Proceedings of the International Workshop on Medical Imaging and Augmented Reality (MIAR '01)*, IEEE Computer Society, Washington, DC, USA, p. 298.

Ciarlet, P. G. (1988). *Mathematical Elasticity: Three-dimensional elasticity*, Elsevier Science Ltd.

Comas, O., Taylor, Z. A., Allard, J., Ourselin, S., Cotin, S. & Passenger, J. (2008). Efficient nonlinear fem for soft tissue modelling and its gpu implementation within the open source framework sofa, *ISBMS '08: Proceedings of the 4th international symposium on Biomedical Simulation*, Springer-Verlag, Berlin, Heidelberg, pp. 28–39.

Cotin, S., Delingette, H. & Ayache, N. (1996). Real time volumetric deformable models for surgery simulation, *VBC*, pp. 535–540.

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

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

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

De, S., Lim, Y.-J., Manivannan, M. & Srinivasan, M. A. (2006). Physically realistic virtual surgery using the point-associated finite field (paff) approach, *Presence: Teleoper. Virtual Environ.* **15**(3): 294–308.

Debunne, G., Desbrun, M., Cani, M.-P. & Barr, A. H. (2001). Dynamic real-time deformations using space & time adaptive sampling, *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, pp. 31–36.

Delingette, H. & Ayache, N. (2005). Hepatic surgery simulation, *Commun. ACM* **48**(2): 31–36.

Deo, D. & De, S. (2009). Phyness: A physics-driven neural networks-based surgery simulation system with force feedback, *World Haptics Conference* **0**: 30–34.

Filipovič, J., Peterlík, I. & Matyska, L. (2009). On-line precomputation algorithm for real-time haptic interaction with non-linear deformable bodies, *Proceedings of The Third Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, pp. 24–29.

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

Gosline, A. H., Salcudean, S. E. & Yan, J. (2004). Haptic simulation of linear elastic media with fluid pockets, *Haptic Interfaces for Virtual Environment and Teleoperator Systems, International Symposium on* **0**: 266–271.

Hager, W. W. (1989). Updating the inverse of a matrix, *SIAM Rev.* **31**(2): 221–239.

James, D. & Pai, D. (2002). Real time simulation of multizone elastokinematic models, *International Conference on Robotics and Automation*, Washington, D.C., USA, pp. 927–932.

J.T.Oden (1972). *Finite Elements of Non-linear Continua*, McGraw-Hill.

Křenek, A. (2003). Haptic rendering of complex force fields, *EGVE '03: Proceedings of the workshop on Virtual environments 2003*, ACM, pp. 231–239.

Miller, K., Joldes, G., Lance, D. & Wittek, A. (2007). Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation, *Communications in Numerical Methods in Engineering* **23**(2): 121–134.

Misra, S., Okamura, A. M. & Ramesh, K. T. (2007). Force feedback is noticeably different for linear versus nonlinear elastic tissue models, *WHC '07: Proceedings of the Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, IEEE Computer Society, Washington, DC, USA, pp. 519–524.

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

Peterlík, I. (2008). Efficient precomputation of configuration space for haptic deformation modeling, *Proceedings of Conference on Human System Interactions*, IEEE Xplore, pp. 225–230. best paper award.

Peterlík, I. (2009). *Haptic Interaction with non-linear deformable objects*, PhD thesis, Masaryk University.

Peterlík, I. & Filipovič, J. (2010). Distributed construction of configuration spaces for real-time haptic deformation modeling, *IEEE Transactions on Industrial Electronics* p. to appear.

Peterlík, I. & Matyska, L. (2008). Haptic interaction with soft tissues based on state-space approximation, *EuroHaptics '08: Proceedings of the 6th international conference on Haptics*, Springer-Verlag, Berlin, Heidelberg, pp. 886–895.

Peterlík, I., Sedef, M., Basdogan, C. & Matyska, L. (2010). Real-time visio-haptic interaction with static soft tissue models having geometric and material nonlinearity, *Computers & Graphics* p. to appear.

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

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

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

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

P.R.Amestoy, I.S.Duff & L'Excellent, J.-Y. (2000). Multifrontal parallel distributed symmetric and unsymmetric solvers, *Comput. Methods in Appl. Mech. Eng.* **184**: 501–520.

Saupin, G., Duriez, C., Cotin, S. & Grisoni, L. (2008). Efficient contact modeling using compliance warping, *Computer Graphics International Conference (CGI) Istambul, Turkey,*.

Sedef, M., Samur, E. & Basdogan, C. (2006). Real-time finite-element simulation of linear viscoelastic tissue behavior based on experimental data, *IEEE Comput. Graph. Appl.* **26**(6): 58–68.

Taylor, M., Cheng, M. & Ourselin, S. (2007). Real-time nonlinear finite element analysis for surgical simulation using graphics processing units, *Medical Image Computing & Computer-Assisted Intervention Conference*, pp. 701–708.

Wriggers, P. (2008). *Nonlinear Finite Element Methods*, 2008 Springer Verlag.

Wu, X., Downes, M. S., Goktekin, T. & Tendick, F. (2001). Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes, *in* A. Chalmers & T.-M. Rhyne (eds), *EG 2001 Proceedings*, Vol. 20(3), Blackwell Publishing, pp. 349–358.

Wu, X., Goktekin, T. & Tendick, F. (2004). An interactive parallel multigrid fem simulator, *ISMS*, pp. 124–133.

Wu, X. & Tendick, F. (2004). Multigrid integration for interactive deformable body simulation, *International Symposium on Medical Simulation (2004).Association for Computing Machinery, Inc*, pp. 92–104.

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

Zhuang, Y. & Canny, J. (1999). Real-time simulation of physically realistic global deformation, *IEEE Vis'99 Late Breaking Hot Topics*.

Zhuang, Y. & Canny, J. (2000). Real-time global deformations, *The fourth International Workshop on Algorithmic Foundations of Robotics (WAFR)*, A. K. Peters, pp. 97–107.

Advances in Haptics

Edited by Mehrdad Hosseini Zadeh

ISBN 978-953-307-093-3
Hard cover, 722 pages
Publisher InTech
Published online 01, April, 2010
Published in print edition April, 2010

Haptic interfaces are divided into two main categories: force feedback and tactile. Force feedback interfaces are used to explore and modify remote/virtual objects in three physical dimensions in applications including computer-aided design, computer-assisted surgery, and computer-aided assembly. Tactile interfaces deal with surface properties such as roughness, smoothness, and temperature. Haptic research is intrinsically multi-disciplinary, incorporating computer science/engineering, control, robotics, psychophysics, and human motor control. By extending the scope of research in haptics, advances can be achieved in existing applications such as computer-aided design (CAD), tele-surgery, rehabilitation, scientific visualization, robot-assisted surgery, authentication, and graphical user interfaces (GUI), to name a few. Advances in Haptics presents a number of recent contributions to the field of haptics. Authors from around the world present the results of their research on various issues in the field of haptics.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen