Doctoral Thesis

Haptic rendering of frictional tool-tissue contact

Author(s):
Leškovský, Peter

Publication Date:
2008

Permanent Link:
https://doi.org/10.3929/ethz-a-005596205

Rights / License:
In Copyright - Non-Commercial Use Permitted
Diss. ETH No. 17532

Haptic rendering of frictional tool-tissue contact

A dissertation submitted to the
ETH ZURICH

for the degree of
Doctor of Sciences

presented by
PETER LESKOVSKY
Mgr., Comenius University in Bratislava

8. Feb. 1979

citizen of Slovakia

accepted on the recommendation of
Prof. Dr. Gabor Szekely, examiner
Prof. Dr. Abderrahmane Kheddar, co-examiner
PD Dr. Matthias Harders, co-examiner

2007
Abstract

The sensation of interaction forces in virtual reality applications enhances the immersion of users in computer generated environments. It helps them to move objects embedded in the virtual environments in a natural way by using all the input stimuli they would get from the real world. The display of contact forces has a big potential in virtual reality based training systems where high dexterity of tool manipulation is required. These are for example surgical simulators.

Providing haptic feedback in surgical simulators is a challenging task, for which the main bottleneck is the computational power of current hardware. In general, a refresh rate of 1 kHz is required for a stable force feedback. This is hard to achieve in virtual environments which model the interaction with deformable bodies. On one hand it is due to the high number of degrees of freedom needed for the representation of the soft body, on the other hand it is due to a possibly large number of contact points acting at a single moment. In the existing simulators, the force output has been therefore often simplified, modeling contacts with single points and with the assumption of frictionless contact.

In this thesis we present two methods for a realistic 6-Degree-Of-Freedom force feedback, considering frictional contact between rigid tools and deformable bodies. The first one is an extension of the popular virtual proxy point method, applied to multiple contacts. The contact points are treated independently, which promises low complexity of the algorithm. Nevertheless, it is based on penalty forces, which can lead to false behaviour under high loads. The second method follows a constraint based contact formulation, which provides physically more precise contact forces than the penalty based approach. Nevertheless, it is computationally more expensive than the first approach, for what it is hard to compensate in real-time. Therefore, in this method we focus on the development of an efficient multirate algorithm, which decouples the full update of the contact forces from their display on the haptic device. A computationally effective haptic rendering algorithm is achieved by linearising the inverse of the contact Jacobians in the active space.

To analyse the fidelity of the provided haptic feedback, we propose a Turing like test, where the participants have to discern blindly between the groups of real objects and virtually generated ones. Our pilot studies have been simple in the sense that
it allowed the user to push on the sample only at one point. Nevertheless, they proved that in our setup we achieved high realism when haptically presenting soft virtual objects, with the stiffness range of soft tissues, to the users. As a result, the relative stiffnesses of the virtual samples have been truthfully recovered by all the participants. Moreover, we observed, that if little noise is present in the haptic output, even if being hardly detectable by the users, it unconsciously amplifies the perceptual difference between the real and the virtual objects. Finally, the results of these experiments identify the limited stiffness of our haptic hardware and the not tuned dynamic properties of the virtually generated objects as the main indicators upon which it is possible to discern between the real and virtual objects.
Zusammenfassung


In dieser Arbeit präsentieren wir zwei Methoden für eine realistische Kraftrückführung mit 6 Freiheitsgraden, welche die Reibung bei Kontakt zwischen starren und deformierbaren Körpern miteinbezieht. Der erste Ansatz ist eine Erweiterung der beliebten virtuellen Proxypunkt-Methode, angewandt auf mehrere Kontaktstellen. Die Kontaktstellen werden hierbei unabhängig voneinander behandelt, was eine geringe Komplexität des Algorithmus verspricht. Der Ansatz basiert auf Strafkraften, die jedoch bei zu hoher Last zu falschem Verhalten führen können. Die zweite Methode verwendet eine Kontaktformulierung basierend auf Nebenbedingungen, die physikalisch präzisere Kontaktkräfte als der Strafkraftansatz bieten. Jedoch ist dieser Ansatz rechnerisch teurer, was eine Realisierung in Echtzeit erschwert. Daher liegt bei dieser Methode der Schwerpunkt in der Entwicklung eines effizienten mehrfachraten Algorithmus, welcher die vollständige Aktualisierung der Kontaktkräfte unabhängig von der Anzeige auf dem haptischen Gerät durchführt. Ein rechnerisch
effektiver Algorithmus zur haptischen Wiedergabe wird durch eine Linearisierung der inversen Kontakt-Jacobideterminante innerhalb des aktiven Raums erreicht.

4.1 Issues of a successful haptic simulator ........................................ 60
4.2 System design ........................................................................ 61
4.3 The implementation basis .......................................................... 63
4.4 Components ........................................................................... 63
  4.4.1 Mass-Spring system (MSS) .................................................. 63
  4.4.2 Parameters setting of the Mass-Spring system ...................... 67
  4.4.3 Numerical integration .......................................................... 72
  4.4.4 Collision detection ............................................................... 79
  4.4.5 Collision response ............................................................... 84

5 Haptic rendering ........................................................................ 87
  5.1 Virtual proxy paradigm ............................................................ 88
  5.2 Virtual coupling ..................................................................... 89
  5.3 Single contact point ............................................................... 90
    5.3.1 Quasi-static proxy point model ........................................ 90
    5.3.2 Smoothing the force output ............................................. 91
    5.3.3 Surface friction .............................................................. 92
  5.4 Multiple contact points .......................................................... 99
    5.4.1 Quasi-static simulation of the proxy object ..................... 99
    5.4.2 Dynamic simulation of the proxy object ......................... 101
    5.4.3 Summary of the penalty based haptic rendering algorithm . 103
    5.4.4 Setting the parameters of the penalty based haptic rendering algorithm .................................................. 104
  5.5 Constraint based formulation of the contact forces .................... 104
    5.5.1 General overview ............................................................ 105
    5.5.2 Contact-free dynamics of the virtual bodies ..................... 107
    5.5.3 Contact handling ............................................................ 109
    5.5.4 Linear contact model ....................................................... 112
    5.5.5 Limitations .................................................................. 116
  5.6 Comparison of the contact force computation models ............... 117

6 Evaluation methods .................................................................. 121
  6.1 Related work ......................................................................... 122
  6.2 System setup ......................................................................... 124
    6.2.1 Computational model ..................................................... 124
    6.2.2 Reference silicone samples ............................................. 124
    6.2.3 Hardware setup ............................................................... 126
  6.3 Discrimination task ............................................................... 126
    6.3.1 Reference samples and virtual objects ............................. 126
    6.3.2 Interaction overview ...................................................... 127
    6.3.3 Experimental apparatus ............................................... 128
    6.3.4 Experimental procedure ................................................. 129
1

Introduction

1.1 Definitions

Touch, as one of our senses, is an inherent part of our everyday life. It can be considered as one of the most important senses in that nearly all life forms have a response to being touched, while only a subset have sight and hearing. The functionality of our touch sense is also unique, since it is not only gathering the information from the environment, but also guiding the movements of our body, thus having a direct effect on the perceived world.

Touch, to be more specific, refers to cutaneous sensations, i.e. perception of phenomena arising from the skin contact with an object. These are for example a sensation of pressure (hence shape, softness, texture, etc.), temperature or pain. The cutaneous sensation is a part of the somatosensory system, which also includes the perception of movements and postures of our limbs, called kinesthesia or proprioception. While touch provides sensation of fine level object properties (e.g. fine
shape, texture, local compliance, etc.), the kinesthetic system conveys coarse scale attributes, i.e. those which have to be explored by controlled motion of our limbs (e.g. large shapes, mass, etc.). Nevertheless, both of these systems provide a sensation of forces stemming from contact.

The word \textit{haptics}, coming from the Greek “\textit{haptesthai}” (to touch), refers to the ability of experiencing the environment through active exploration, for which cutaneous and kinesthetic capabilities are important. In this work we will use special hardware, robotic machines, to present contact forces from a virtual simulation to the user, thus enabling the user to touch, feel and manipulate virtual objects. The computation and display of forces to a user via haptic devices is referred to as \textit{haptic rendering}.

\subsection{1.2 Haptic devices}

In the last decades, devices capable of generating different kinds of touch stimuli, so-called \textit{haptic} or \textit{force feedback devices}, have been developed in numerous research centres.

Haptic interfaces can be categorised according to the type of response which they provide into kinesthetic and tactile devices. The first kind stimulates the sensors of our muscles and joints by applying forces (e.g. generated by motors).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{common_haptic_devices}
\caption{Common haptic devices: the Omega device by Force Dimensions providing 3-DOF-, the Phantom device by Sensable with 6-DOF- and Xitact IHP generating 4-DOF- force feedback (from left to right).}
\end{figure}

For haptic rendering of virtual worlds, common kinesthetic interfaces are small desktop devices like the Omega, providing three degrees of freedom (3-DOF) force output, and the PHANToM, with 6-DOF force feedback (rendering torques in addition). They are shown in Figure 1.1. These mechanisms are used in applications which model tool interactions with the virtual environment. While the 3-DOF interfaces are best suited for modelling of single point contacts, mostly assuming an interaction
with the tool-tip, the 6-DOF devices can simulate the force feedback gained during
general tool use.

Customised devices have been used for special tasks. For example, the Xitact IHP
(Figure 1.1, right) has been designed for surgical simulators modelling minimally
invasive interventions (see application in [Mentice AB, 2000]). It generates forces in
three rotations around a fixed point and one translation along the main tool axis
and is thus able to provide the same haptic feedback as the surgeons experience
when using long thin instruments inserted through an opening in the patient’s body
in minimally invasive surgery tasks. Additional feedback can be supplied if grasping
or cutting actions are simulated. Another example of a more complex design is the
glove like haptic device Sensor Glove II [Kitada et al., 1997] (Figure 1.2). This 20-
DOF haptic mechanism renders forces at each of the phalanges of the hand (4-DOF
for each finger) and thus allows for simulations of hand motions, especially hand
grasping actions.

![Figure 1.2: Sensor Glove II [Kitada et al., 1997]: a haptic device designed with 20-
DOF force feedback applied at each joint of the hand. The mechanical design (left)
and the glove with attached wires to the pulleys for generating the force response
(right).](image)

The second group of haptic devices, the tactile displays, are used to reproduce
sensations similar to those when we touch objects with the fingertip and slide over
the surface. Devices with miniature movable pins or inflatable bladders are used
to indent or vibrate the skin locally in order to stimulate the cutaneous receptors.
This way, texture of a material or small shapes, like Braille characters, can be
haptically rendered. In Figure 1.3, left, a typical pin–based display, developed at
the Forschungszentrum Karlsruhe, is shown. The miniature pins, set densely onto
a small area, can independently indent the user’s fingertip and thus model rough
spatial approximations of textures. Another example of a tactile display, based on
different principles of generating tactile feedback, is the STReSS² device (Figure 1.3,
right). It uses lateral vibrations to create an illusion of movement on the skin
(an illusion also known as the comb illusion) [Wang and Hayward, 2006]. The
combination of kinesthetic and tactile haptic devices promises an advanced solution for simulating free hand interaction in virtual environments and palpation of virtual bodies.

Figure 1.3: 2D tactile displays: pin based device with 72 electromagnetic actuators, built at the Forschungszentrum Karlsruhe (left), and STReSS device, built at McGill Haptics Lab, composed of an array of piezoelectric benders (right).

1.3 Haptic rendering of tool-tissue contact

The developments in the haptic field demonstrated a great impact in the simulation of virtual environments where touch and force sensation enhances the interaction with the presented objects. Haptic rendering has been applied to tasks like exploration of complex datasets, high precision remote manipulation of tools, 3D design and modelling, virtual assembly and surgical simulation. Among these, surgical simulation is one of the main application areas where contacts between rigid tools and soft bodies are simulated. The main focus of this thesis is on the modelling and haptic rendering of tool-tissue interactions. Therefore, the contact force computation models, applied in virtual reality based surgical training, as well as the factors which influence the quality of the provided force feedback will be discussed to a large extent.

1.3.1 Surgical simulators

In surgical training, computer based simulators are used to model virtual patients on which prospective surgeons can perform specific surgical tasks or whole medical procedures [Edmond et al., 1997; Székely et al., 2000; Tendick et al., 2000]. Virtual reality based surgical simulators provide an alternative to the traditional approach of learning surgical skills, where training on cadavers, animals and patients is mostly

\footnote{To get an overview of existing virtual reality based surgical simulators, the reader is referred to \url{http://www.virtualsurgery.vision.ee.ethz.ch}.}
practised. These provide an environment where surgical interventions can be taught without danger to patients. What is more, the trainees can be easily introduced to rare and unusual cases. Lately, several commercially available surgical simulators have been presented [Mentice AB, 2000; VEST Systems, 2001; Surgical Science Sweden AB, 2001; Simbionix, 2002; Melerit medical, 2005] and attempts have been made to integrate these systems in surgical training programs.

**Figure 1.4:** The nature of a traditional open surgery (left) and Minimally Invasive Surgery (right).

Two main groups of surgical simulators exist: one modelling the conventional open surgery and the other dealing with Minimally Invasive Surgery (MIS). Open surgery (Figure 1.4, left) is the traditional approach to surgical interventions, where the surgeon has to cut the tissues in order to obtain access to the organs involved in the procedure. To this end, large incisions have to be made, muscles have to be separated from the bones or organs and inner organs must be pushed aside. For the intervention, classical surgical instruments, like scalpels, scissors, hooks and others, are used. When needed, the surgeon also touches and moves the organs with his hands.

In MIS (Figure 1.4, right), small incisions into the body of the patient are made, into which metallic ducts, trocars, are inserted. The trocars are used to introduce miniature instruments, attached at the end of a long rigid shaft, and small cameras into the body cavities. The use of the miniature tools allows the surgeons to operate without the need of making large cuts into the body.

The technique of accessing the inner organs via a few small incisions, represents the main advantages of MIS over open surgery for the patient. Reduced operative trauma and faster recovery are among the main reasons why MIS became popular in the last decade. However, this new surgical approach poses severe difficulties for the surgeons. Due to the non-usual view, obtained via an inserted camera, the hand-eye coordination is the main difficulty to overcome. This task gets even harder due to the reduced mobility of the instruments, restricted by the fixed insertion point.
To train surgeons for the unusual tool manipulation in MIS was the main reason, for the development of surgical trainers. In several MIS simulators, only a static environment has been provided, in which the trainee has to navigate the camera or the surgical tools, in order to expose a certain area, or to reach given positions [Haluck et al., 2001]. This way the surgeons acquired an increased dexterity in the manipulation of the instruments.

To also teach the novice surgeons all particular actions which have to be processed during an intervention, active virtual environments have been constructed. In these, particular tasks like cutting and suturing can be practised [Downes et al., 1998; Kühnapfel et al., 2000; Basdogan et al., 2001; De et al., 2002], or even a whole surgical procedure can be followed [Montgomery et al., 2000; VEST Systems, 2001; Simbionix, 2002; Harders, 2006].

Teaching the key surgical procedures and providing a training scenario for emergency situations is also desirable for the open surgery education. Nevertheless, up to now, only a few simulators modelling open surgical interventions exist [Bro-Nielsen et al., 1998; Webster et al., 2001; Berkley, 2002; Pflesser et al., 2002; Bielser, 2003; Suzuki and Suzuki, 2003].

1.3.2 Haptics as a part of surgical simulators

During surgical interventions, the surgeons rely mainly on two senses: vision and touch. Nevertheless, in many tasks, like the MIS, the haptic feedback is reduced or strongly distorted. The surgeons learned to compensate for the reduction or even for a complete loss of the touch using the visual system. Visible cues like the deformation or the color change of the soft tissues provide them with a valuable information about the forces applied during the surgical task. The daVinci\textsuperscript{R} Surgical System\textsuperscript{2}, used for laparoscopic surgeries, is the best example of the surgeons’ adaptation. In this system the surgical instruments are driven by a robotic platform which is controlled by the surgeons via an interactive computerised interface. While the surgeon controls the motion of all instruments, he only receives a visual feedback acquired by the laparoscopic camera.

Although it is possible to perform surgical tasks without the sense of touch, receiving haptic feedback during surgeries is beneficial for the surgeons. Not only that including the touch sense the motoric tasks become more intuitive, but it also helps the surgeons to discriminate a healthy from an abnormal tissue and to identify parts of organs which are obscured on the camera view. Moreover, it enhances their performance by improving the accuracy and speed of the executed task. This can be observed for expert as well as novice surgeons [Cao et al., 2007]. Similarly, providing

---

\textsuperscript{2}Intuitive Surgical\textsuperscript{R} (http://www.intuitivesurgical.com)
touch feedback during surgical training can contribute to the skill acquisition of the trainees, which is the main reason, why the display of haptic cues became an important part of surgical simulators.

In general, the contact force models used in surgical simulators for providing haptic information during an interaction with the organs have been kept simple. Indeed, the need for high fidelity haptic rendering depends on the specific application area. Several tasks in surgical training exist, where force feedback is of low priority. This is the case of camera navigation in laparoscopic surgery [Haluck et al., 2001], or the development of key psychomotor skills. Also, in some cases low level haptic feedback can still provide valuable input leading to satisfactory surgical task performance [Brouwer, 2004].

Overall, the lack of computational power, but also the specific constraints in MIS training, were the main reasons for introducing simplifications in the modelling of contact forces. In the following, differences between MIS and open surgery simulators will be described which illustrate the need for higher fidelity force feedback in the simulation of conventional surgical tasks.

Basic differences between the open surgery and MIS virtual scenarios are due to the different mechanical properties of the used haptic devices. Since the tools used in MIS are inserted through trocars, the DOF of their movement is usually reduced to four (three rotational DOF around a fixed point and one translational DOF along the trocar). For open surgery, on the contrary, haptic devices with six DOF have to be used in order to model interaction with a tool held in a hand. Moreover, additional DOF are necessary if hand grasping actions are also modelled in the simulator. Besides the difference in DOF, one should also consider, that a larger working space and higher force output may be required during certain conventional surgical tasks.

The primary difference in the quality of haptic feedback between MIS and open surgery simulators is also in the level of forces which can be sensed during the intervention. Since surgeons performing a MIS have their instruments always in contact with the trocar, a considerable part of the force which the surgeons feel stems from the movements of the tool inside the trocar and from the restrictive insertion point of the trocar. Therefore, the perception of the force, originating from interaction with organs during a MIS, is substantially reduced [Xin et al., 2006]. What is more, the customised handles of the surgical tools only emphasise this decrease. In a traditional open surgery, the surgeons hold the instruments in their hands directly and operate with them in the vicinity of organs. This, in contrast to MIS, allows for better perception of forces applied between the tools and the organs. This was also justified by Gupta et al. [1996], who observed that, in contrast to open surgery, the tip forces in MIS are not efficiently translated to the handle forces.
1. Introduction

Simulation of tool interactions in MIS simulators was also simplified due to the type of the surgical instruments used and their manipulation style. Long, thin tools are used, for which the interaction can be reduced to contacts with the tool–tip (i.e. with the miniature instrument attached at the tool–tip) and the shaft of the tool. This makes a simulation of touching soft tissues using only point- [Ho et al., 1999] or ray-based models [Basdogan et al., 2000] of the virtual tools appropriate. For open surgery, it is more important to detect multiple contacts between a complex tool and the tissue, since a higher quality 6-DOF force response has to be generated. An interaction with the whole surface of the virtual tool, should be considered. Nevertheless, this requirement poses high computational demands on today’s hardware, which is due to the complexity of general collision detection between the tool and a deformable body as well as the nontrivial contact force calculation. Therefore, to facilitate these issues, simplified tools, composed of a set of line segments have commonly been modelled in existing surgical training systems [Basdogan et al., 2001; Bielser, 2003].

Only little attention has been paid to a physically correct calculation of the contact force and contact phenomena like sticking or sliding. The modelling of friction has either been neglected, or a full stiction at the contact point has been applied. Only lately a Coulomb friction model, modelling stiction as well as dynamic friction, has become the common approach for describing frictional contact. The absence of proper friction force generation in tool tissue contacts makes certain manipulation tasks of the virtual objects difficult or even impossible, thus leading to inability of providing adequate training scenarios. One of such scenarios, where frictional effects play an important role, is cutting of a floating piece of tissue during a hysteroscopy intervention. To achieve this task, the surgeon has to push the tissue against the wall of the uterine cavity. If no friction between the tool, the tissue and the wall is applied, the floating piece slips away from the tool when touching it lightly. With full stiction the task would become trivial, but also very unrealistic. Clearly, proper frictional characteristics have to be set to allow for addressing this task.

1.3.3 Fidelity of the haptic feedback

Haptic feedback is important for our daily life. The consequences of a major loss of the somatic sense were documented in [Robles-De-La-Torre, 2006] in several patient studies. It is concluded that:

\textit{It results in catastrophic impairments of hand dexterity, haptic capabilities, walking, perception of limb position, and so on.}

The author further points out, that a loss of the somatosensory system can not be adequately compensated by any other of the human senses. The influence of haptic feedback on the performance in virtual environments can be similar:
Providing users with inadequate somesthetic feedback in virtual environments might impair their performance, just as major somesthetic loss does. [Robles-De-La-Torre, 2006]

A central element of surgical simulation is therefore the generation of adequate haptic feedback. Several factors influence the force rendering process, which could potentially degrade the feedback quality. As a consequence, the trainee’s performance in the virtual environment can be reduced, leading to low training effectiveness. With regard to this, it is essential to identify the limitations of the applied haptic rendering approaches. These are introduced by the selected hardware (i.e. the haptic device) as well as the simplifications applied in the haptic rendering algorithms. One option to approach this issue is to compare the interaction modelled in the virtual environment to a real world setting. Such studies would provide better understanding of the perception of virtually generated haptic stimuli. Moreover, the results will be helpful for the identification of important sources of haptic information as well as the estimation of the degree of fidelity needed when providing haptic feedback to the user.

1.4 Overview

In this thesis we focus on realistic real-time contact modelling and haptic rendering of interaction between virtual deformable objects and simple rigid tools. The main emphasis will be put on the technical view of contact force modelling as well as on computational efficiency for achieving real-time haptic rendering.

We will first discuss about a design of a simplified virtual environment, the haptic simulator, where several contact force computation approaches will be tested for haptic rendering. The interaction within our virtual environment will be restricted to noninvasive applications like pushing, pulling and sliding over the surface of the flexible body, while providing realistic 6-DOF haptic feedback.

The core part of our virtual environment is the simulation of the deformations of a flexible body. A real-time performance and a realistic parametrisation of the deformation model is essential for the generation of realistic haptic feedback. Setting up a simple, yet credible, deformation model for the soft body will thus be the first issue during the implementation.

Besides the interaction force stemming from the deformation model, we put the emphasis on modelling frictional forces when touching flexible bodies. A Coulomb’s based friction model, distinguishing between static and dynamic friction, will be included in the haptic rendering algorithms. This will allow the user to sense the stick and slip phenomena, characteristic to dry contact.
During the implementation, necessary simplifications of the object’s response, such as the deformation and contact modelling, are made in order to achieve real-time interaction speed with complex objects. Since the introduced simplifications could diminish the quality of the haptic response, an evaluation of the generated haptic feedback is necessary. Although several evaluation techniques have been proposed, these have been mainly designed specifically for a given component of the haptic rendering pipeline (i.e. the deformation model, the haptic rendering algorithm, the force smoothing algorithm, the haptic interface, etc.). Only a few projects have been carried out, where the fidelity of the force output, as it is perceived by the users, has been addressed. Nevertheless, none of the existing evaluations explored minutely the degree of realism which was perceived by the user when touching deformable objects with a tool. In this sense, we will present a technique quantitatively measuring the level of realism achieved within a virtual haptic interaction. A simple virtual scenario will be set against physical interaction with real soft bodies in order to assess the fidelity of the rendering approach.

The advantage of low computational complexity while providing realistic sensation of the contact force makes the investigated approaches suitable for open as well as minimally invasive surgical training systems. High quality haptic rendering within a surgical simulator is therefore the main application area of our studies.

The main points of interest in this thesis can be summarised to:

- 6-DOF haptic contact force modelling between rigid tools and deformable objects, employing the haptic rendering of multiple contact points
- modelling of frictional phenomena; showing different behaviour of the objects when subject to frictional contact
- evaluation of the realism for a simplified contact model
- application in surgery simulators and virtual reality environments.

1.5 Structure of the thesis

In the next chapter we provide a short introduction into the physical background of contact force computation. Contacts between deformable bodies and rigid tools, including frictional effects, will be considered.

In Chapter 3, we will describe the state of the art of haptic interaction with virtual soft bodies. Two important parts of the contact force computation will be described in more detail. First, models simulating the deformation of elastic bodies will be presented. Then several methods for computing the contact forces between collided
objects in a virtual environment will be described. We complete the overview by presenting approaches of applying the contact force models to haptic rendering.

After the introduction into the problemsatics of contact force computation, Chapter 4 will describe our simulation framework. A multithreaded application with its components will be introduced. Special attention will be given to the used deformable model, parameter setting procedures, numerical integration of the dynamic system and collision detection as well as response algorithms.

More details of haptic force generation and rendering will be given in Chapter 5. Force rendering based on single point and multiple points contact, will be described. Two different force response strategies will be presented – a penalty based and a constraint based contact force computation. For each technique, the applied smoothing techniques and utilised friction models will be mentioned. At last, a comparison of the two proposed models will be given.

An evaluation of the force feedback realism, based on a comparison of interaction with real and virtual objects, inspired by the idea of a Turing test, will be presented in Chapter 6.

Three main application directions will be shown in Chapter 7. The first application presents our general framework, in which several scenarios will be implemented in order to present realistic force feedback with different frictional properties. In addition, the developed techniques will be also presented in the context of a hysteroscopy simulator and an augmented reality setup.

Our main results in contact force computation and a novel evaluation technique will be summarised afterwards, in Chapter 8.
2

Physical foundations

In this thesis we will study frictional contact between an elastic solid and a rigid body. To introduce the reader into the problematics of the contact force computation we now briefly describe the physical principles of the contact phenomenon.

2.1 Preliminaries

The contact problem in three dimensional space can be stated as follows. Consider a deformable body $\mathcal{B}_D$ as a regular region $\Omega_D \subset \mathbb{R}^3$ with surface $\Gamma_D$. Let the deformation of the elastic body be expressed by a configuration mapping $\Phi_D^t : \Omega_D \to \mathbb{R}^3$, which maps each point in material coordinates, defined in the rest state, to its position in 3D space at time $t$. It is common to express the configuration $\Phi_D^t$ as a combination of the identity mapping $\text{Id}$ and a displacement field $u$: $\Phi_D^t = \text{Id} + u^t$.

Similar, the state of a rigid body $\mathcal{B}_R$ can be described by a configuration mapping $\Phi_R^t : \Omega_R \to \mathbb{R}^3$. Nevertheless, for the rigid body the configurations are restricted to rigid body motion, i.e. $\Phi_R$ is defined by rotation and translation.

![Figure 2.1: Elastic body $\mathcal{B}_D$ in contact with a rigid tool $\mathcal{B}_R$.](image)
We will partition the boundary $\Gamma_D$ of the deformable specimen into three disjoint measurable subsets: $\Gamma_1$, $\Gamma_2$ and $\Gamma_3$ (Figure 2.1). On $\Gamma_1$ the body is fixed, thus the displacement field vanishes: $x \in \Gamma_1 : u(x) = 0$. $\Gamma_2$ represents a contact free area on which surface tractions of density $\varphi_S$ are acting. Finally, the contact between the rigid tool $B_R$ and the elastic body will be defined on the portion $\Gamma_3$.

The behaviour of the elastic body can be described by the following partial differential equations (see also [Wriggers, 2002]):

\[
\begin{align*}
\text{div} \, \sigma + f_B &= \rho \ddot{u} \quad \text{on } \Omega \text{ (equation of equilibrium)}, \\
\text{div}(\rho \dot{u}) + \dot{\rho} &= 0 \quad \text{on } \Omega \text{ (conservation of mass)}, \\
\sigma &= E(\varepsilon(u)) \quad \text{on } \Omega \text{ (constitutive law of elasticity)}, \\
\varepsilon &= B(u) \quad \text{on } \Omega \text{ (strain formulation)}, \\
u &= 0 \quad \text{on } \Gamma_1 \text{ (fixed boundary)}, \\
\sigma \mathbf{n} &= \varphi_S \quad \text{on } \Gamma_2 \text{ (contact free boundary)}, \\
\sigma &= F(\Phi_D, \Phi_R) \quad \text{on } \Gamma_3 \text{ (contact area)}.
\end{align*}
\]

Here, $\sigma$ is the Cauchy stress tensor, $\varepsilon$ is the strain tensor, $f_B$ is a vector of body forces (i.e. gravitational forces, etc.), $\rho$ is the material density, $\mathbf{n}$ denotes the unit outer normal defined at the surface $\Gamma_D$ and $F(.)$ is a force function describing the contact between the two solids.

The Equation 2.3, which relates the material stress to the strain, is dependent on the elastic properties of the material. Although the following physical fundamentals will be formulated regardless of the material of the deformable body, in our later study we will examine homogeneous and isotropic linear elastic materials. For such, a constitutive law based on the Hooke’s law can be used:

\[\sigma = 2 \mu \varepsilon + \lambda \text{tr}(\varepsilon) I,\]

where $\mu$ and $\lambda$ are the Lamé coefficients, and $I$ is the $3 \times 3$ identity matrix. The relation between the strain and the displacement field (see Equation 2.4) can be expressed by the Green-Lagrange strain tensor:

\[\varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \frac{1}{2} \left( \frac{\partial u_k}{\partial x_j} \frac{\partial u_k}{\partial x_i} \right).\]

For small displacements this form can be reduced to the Cauchy’s strain tensor by neglecting the second, nonlinear part.

For practical solutions, the Equations 2.1,2.2,2.3 and 2.4 can be discretised over the domain of the elastic body, resulting in a system of ordinary differential equations:

\[M \ddot{u} + D \dot{u} + K(u) = F_{\text{ext}}.\]
Here, $M$ represents the object’s mass matrix, $D$ its damping matrix, $K(\cdot)$ is a function of the internal forces (given by Equation 2.3) and $F_{\text{ext}}$ represents the external force field. The last equation can be also expressed in quasi-static manner, thus neglecting the inertial and velocity terms and solving for the state of the soft body at a static equilibrium:

$$K(u) = F_{\text{ext}}.$$  \hspace{1cm} (2.9)

Given the boundary conditions expressed in Equations 2.5, 2.6 and 2.7, the solution of the Equations 2.8 or 2.9 can be obtained numerically.

### 2.2 Point contact model

Of a particular interest is the contact force function $F(\Phi_D, \Phi_R)$, which depends on the configuration of the deformable as well as the rigid body. For further explanations, let us first decompose the contact tractions into normal and tangential forces, $F_n$ and $F_t$ respectively:

$$F_n = (F(\Phi_D, \Phi_R) \cdot n) \cdot n; \quad F_t = F(\Phi_D, \Phi_R) - F_n.$$  \hspace{1cm} (2.10)

We will begin with an evaluation of the contact forces in the normal direction, considering frictionless contact.

#### 2.2.1 Frictionless contact

The contact phenomenon can be described by studying a pair of points $x_D$ and $x_R$, set at the surface of the elastic and the rigid body, respectively, which are the candidate particles for contact. A gap $g = (x_R - x_D) \cdot n$, which is the signed distance between these two points, is used to measure the interaction force between the collided objects (Figure 2.2).

![Figure 2.2: The gap between the two nearest points $x_D$ and $x_R$, i.e., the candidates for contact. Picture taken from [Renaud and Feng, 2003].](image)

The formulation of an unilateral contact problem assumes two conditions:
• impenetrability,
• no attraction forces.

The first requirement means that the particles of both objects do not cross the boundaries of the other body. This feature is expressed by restricting the gap $g$ to non-negative values. The second limitation states, that only repulsive forces act between the objects during the contact event. Thus, the reaction force in the normal direction, $F_n$, is also non-negative and it vanishes when the bodies are not in contact. Although it is possible to also consider adhesion, in this work we will not investigate the influence of attractive forces on the contact phenomena.

The relations of the unilateral contact can be formulated in complementary equations, commonly known as the Signorini conditions:

$$
g \geq 0, \quad F_n \geq 0, \quad g F_n = 0. \quad (2.11)$$

The last formula expresses the fact, that the reaction force exists only when the bodies are in contact. The static contact problem with Signorini’s contact conditions has been discussed for example in [Jourdan et al., 1998; Duriez et al., 2004; Renouf et al., 2005].

Another way to estimate the contact forces is so-called normal compliance law. In this model the bodies are allowed to penetrate each other to a small extent. The forces are then a function of the penetration depth or volume. Usually a power law is used:

$$F_n = c g^m, \quad (2.12)$$

where the constants $c$ and $m$ are determined empirically. Examples of the use of the penalty based contact force computation can be found in [Andersson, 1991; Fernández-Garcia et al., 2000]. In principle, the normal compliance law can be seen as a relaxation of the stiff Signorini conditions, taking into account the gap violation. Physically this model is explained by considering elastic properties of the asperities covering the surfaces of the contacting bodies.

The equations described so far, give the rules according to which an inelastic contact between two bodies can be simulated. These include resting contacts or contacts where the impact energy is consumed by the deformation of the specimen. For elastic contacts, where the bodies do not keep touching each other after the collision, but rather bounce away, additional principles are used. A shock law, e. g. Newton’s non-smooth impact law ($v_n^+ = -e v_n^-$), which relates the velocities before and after the impact in normal direction ($v_n^-$ and $v_n^+$, respectively), is applied. The constant $e$ represents a restitution coefficient which can be measured experimentally. It compensates for the dissipation of the kinetic energy of the system due to permanent
deformation of the specimen, generated heat, sound, etc. Values from zero to one are possible, corresponding to fully inelastic and pure elastic impacts, respectively. During a surgery slow movements and controlled load are applied by the surgeons when interacting with the organs. Thus, the applied contacts are of the inelastic kind \((e = 0)\). Due to this fact, we will not include the shock law into the description of the contact problem.

### 2.2.2 Including frictional effects

Friction is the tangential reaction force between two surfaces in contact. Although it results from many different mechanisms, it always acts in a direction which prevents the relative tangential movement of the two surfaces, i.e. in the direction opposite the sliding velocity. The friction force depends on many parameters like the contact geometry, properties of the surface material (including roughness, wear, etc.), temperature and presence of lubricants. Besides these, the friction characteristics has been explained by considering the relative velocity as well as displacement of the bodies.

In general two modes of friction can be considered, the static friction and the dynamic friction. In the static mode, called also stiction, some friction force may be exerted without sliding, i.e. without any relative tangential movement. However, after overcoming some threshold, sliding will occur and dynamic friction will be applied. This form of friction is referred to as dry friction and it is the most common model used for tool interactions. The dynamic friction regime can be influenced by added lubricants, which introduce viscous characteristics to the friction model.

To describe the tangential contact problem, different friction models can be used. We will now shortly introduce several of them, however, for an extensive survey addressing issues of friction modelling and control the reader should consult [Armstrong-Héloüvry et al., 1994; Olsson et al., 1998]. To simplify the notation, in the following we will consider the tangential contact space to be one-dimensional.

A popular friction model is the non-smooth friction characteristics developed by Coulomb (1785). The friction force is defined proportional to the normal load \(F_n\), i.e. \(F_C = \mu F_n\), where the coefficient \(\mu\) depends on the material properties of the bodies in contact. It is applied opposite to the relative tangential motion \(v_t\). At zero velocity, the friction is not specified by the Coulomb’s model. It is a function of the external shear force \(F_E\) and it can take any values between \(-F_C\) and \(F_C\). The Coulomb’s friction model can be summarised in the following equations:

\[
F_t = \begin{cases} 
-\min(F_E, F_C) \text{sgn}(F_E) & \text{if } v_t = 0, \\
-F_C \text{sgn}(v_t) & \text{if } v_t \neq 0.
\end{cases}
\] (2.13)

Note, that the function \(\text{sgn}(\lambda)\), returning the sign of the variable \(\lambda\), is to be replaced with the normalised vector of the variable \(\lambda\) for higher dimensional spaces, i.e. \(\frac{\lambda}{\|\lambda\|}\).
Considering lubricated surfaces, viscous friction can be added to the dynamic mode of Coulomb’s friction:

$$F_t = -F_C \text{sgn}(v_t) - bF_n |v_t|^{\delta} \text{sgn}(v_t), \quad v_t \neq 0,$$

with $b$ and $\delta$ parametrising the viscous characteristics of the friction force. A linear viscous friction model combined with the Coulomb’s model is shown in Figure 2.3 b).

In many experiments it has been observed, that the friction force at rest can be higher than the friction level applied while sliding, showing the need of higher shear force in order to initiate slip. This leads to the so called stick-slip effect of the dry friction, describing the sometimes jerky or intermittent motion of sliding objects. Stick-slip oscillations can be modelled by using different friction coefficients, $\mu_S$ and $\mu_D$ ($\mu_S \geq \mu_D$), for the static ($v_t = 0$) and for the dynamic regime ($v_t \neq 0$) of the friction force, respectively:

$$F_t = \begin{cases} 
  -\min(F_E, F_S) \text{sgn}(F_E) & \text{if } v_t = 0, \\
  -F_D \text{sgn}(v_t) - F_v |v_t|^{\delta} \text{sgn}(v_t) & \text{if } v_t \neq 0,
\end{cases}$$

for $F_S = \mu_S F_n$, $F_D = \mu_D F_n$ and $F_v = bF_n$ expressing the static, dynamic (or Coulomb’s) and viscous friction, respectively. A friction model adapted for the stick-slip phenomenon is shown in Figure 2.3 c). Variations of this stick-slip and viscous friction include the modified Karnopp’s model, proposed in [Richard, 2000], where the static friction is applied also for small non-zero velocities, i.e. $0 < |v_t| < v_{\text{static}}$. 

\[\text{Figure 2.3: Examples of friction models. a) Coulomb’s friction; b) Coulomb’s friction model with added viscous friction; c) modelling different levels of static and dynamic friction; d) including of the Stribeck effect. Picture taken from [Olsson et al., 1998].}\]
This choice overcomes the difficulties of accurate computations of the objects’ velocities at low speeds, which are inherent for all discrete time simulations.

It has been also found, that the friction force does not decrease discontinuously from the static to the dynamic friction. Instead, a nonlinear dependency of the frictional coefficient on the relative velocity has been determined. This friction characteristic is called the Strubeck effect. The typical friction curve of this model is shown in Figure 2.3 d) and can be expressed by:

\[
F_t = \begin{cases} 
-\min(F_E, F_S) \text{sgn}(F_E) & \text{if } v_t = 0, \\
-F_d + (F_S - F_D) e^{-\frac{v_t}{v_S} |\delta_S|} \text{sgn}(v_t) - F_e |v_t|^{\delta} \text{sgn}(v_t) & \text{if } v_t \neq 0, 
\end{cases}
\]

(2.16)

where \(v_S > 0\) is the Strubeck velocity and \(\delta_S\) the shaping parameter of the Strubeck curve.

The models presented so far differentiate between two friction modes, i.e. static and dynamic, based on the relative velocity. In the static case, applying force smaller than the stiction results in no relative motion. Nevertheless, during the sticking regime, small displacements of the bodies have been observed [Courtney-Pratt and Eisner, 1957]. This microscopic motion arises with tangential compliance and since there is no true sliding, it is called pre-sliding displacement (Figure 2.4). This phenomenon is present for small shear forces, which produce elastic deformation and movement. Releasing the applied force the displacement either vanishes, thus modelling an elastic connection between the surfaces, or it results in permanent displacement, indicating that sliding occurred.

Figure 2.4: The relation between friction and displacement as found by Courtney-Pratt and Eisner [1957]. The dashed lines show the behaviour when releasing the applied force. The elastic portion of the measured displacement refers to pre-sliding displacement.

Dahl [1968] was the first who systematically studied the pre-sliding displacement. In his model, the origin of friction is in spring-like behaviour of the connections
between the surface asperities. The model behaves as a brush whose bristles must be bent as the brush moves in one direction and then flop or bend in the opposite direction if the motion is reversed. Dahl’s friction model can be expressed by a differential equation:

\[
\frac{dF_t}{dx_t} = \sigma_0 \left(1 - \frac{F_t}{F_C} \text{sgn}(v_t)\right)^i, \quad \sigma_0 > 0,
\]

(2.17)

where \(F_C\) denotes the Coulomb’s friction force, \(\sigma_0\) is the stiffness coefficient and \(i\) a parameter that specifies the shape of the stress-strain curve. Dahl’s friction model does neither capture stiction, the Strubeck effect, nor a viscous friction. Slight modification of this model, which includes these additional phenomena, can be found in [Hayward and Armstrong, 2000]. Overall, the Dahl’s model is applied mainly for its better accuracy while describing the friction characteristics in the low velocity regimes.

Another model that can be viewed as an extension of the Dahl’s model is the LuGre (Lund Grenoble) model [Canudas et al., 1995]. Similar to the Dahl’s model, the friction force is given by the average deflection force of elastic bristles. The LuGre model can be seen as a first order Dahl model with a velocity-varying coefficient to give stiction. It also incorporates the Strubeck effect and viscous friction, which makes it more suitable for friction modelling of lubricated contacts. The LuGre model is given by:

\[
F_t = \sigma_0 z + \sigma_1(v_t)\dot{z} + \sigma_2 v_t, \quad \sigma_0, \sigma_1, \sigma_2 > 0,
\]

(2.18)

\[
\dot{z} = v_t \left(1 - \frac{\sigma_0}{g(v_t)} \text{sgn}(v_t)\right) z,
\]

(2.19)

where \(\sigma_0\) is the stiffness of the bristles, \(\sigma_1(v_t)\) the damping of the bristles, \(\sigma_2\) the viscous friction parameter and the variable \(z\) describes a dynamically changing state of the friction model, in this case the elastic portion of of the displacement. The function \(g(v_t)\) modells the Stribeck effect, and can be approximated by:

\[
g(v_t) = F_D + (F_S - F_D) e^{-\left|\frac{v_t}{v_S}\right|^2},
\]

(2.20)

with \(F_S\) and \(F_D\) expressing the static and dynamic (or Coulomb’s) friction force and \(v_S\) denoting the Stribeck specific velocity parameter.

The presented friction models give just a basic overview of friction modelling techniques. Nevertheless, the tribological phenomena are much more complicated than outlined in this section. A number of additional friction characteristics can be considered, such as the dependency on various material properties, material wear, contamination, heat generation, motion direction, force rate, addition of adhesion forces, or hydrodynamic effects.
2.3 Area contact problem

Up to now only single point contacts were considered. Such models can be applied to rigid body contact, or to approximate contacts resulting in global deformation, e.g. when bending an elastic beam. However, for deformations localised in a small region around the contact point, which is the case when indenting soft tissue, the response forces depend essentially on the deflection distribution over the contact area.

The force response for a tool contact can be obtained by superposing the stress distribution due to single point contact and integrating the partial force contributions over the whole contact region. This process, however, should also consider the variability of the pressure distribution in the contact area, which depends on the shape of the tool. For simple geometries, i.e. sphere, cylinder, cone, etc., analytical expressions can be found. Relations for the pressure, stress and radial stress distributions and normal and radial displacements for flat cylindrical, spherical and conical indenters are summarised for example in [Fisher-Cripps, 2000]. The force deflection response for rigid sphere, flat cylindrical and cone indenters, as shown on Figure 2.5, being in frictionless contact with an infinite linear elastic soft body with Young’s modulus $E$ and Poisson’s ratio $\nu$ are given by:

- spherical indenter of radius $r$: $F_n = \frac{4}{3} \sqrt{r} \frac{E}{1-\nu^2} d^3$,
- flat cylindrical indenter of radius $r$: $F_n = 2r \frac{E}{1-\nu^2} d$,
- cone indenter with an opening angle $\alpha$: $F_n = \frac{2}{\pi} \tan \alpha \frac{E}{1-\nu^2} d^2$,

with normal contact force $F_n$ and penetration depth $d$.

![Figure 2.5: Examples of simple, axis-symmetric indenters, for which analytical force deflection relations can be found in [Fisher-Cripps, 2000].](image)

To solve the contact problem for more complex geometries, relations leading to elliptical integrals are obtained. For practical solutions the contact space is discretised and the response forces are obtained numerically. More details on this can be found in [Wriggers, 2002].
Considering frictional forces applied over an area contact extends the complexity of the contact phenomenon. Due to the different load distribution and the varying surface properties effects like microslip can occur. This corresponds to states, where in certain parts of the contact region the surfaces slip, whereas in other parts full stiction is present.

Taking micro-mechanical phenomena into account further complicates the contact problem. As an example, the dependence of the pressure distribution on the surface roughness can be examined. Considering that the geometry of a surface at micro-scale is in fact very rough, the contact area is formed by the asperities of the surfaces in contact. Applying higher load causes more asperities to be joined, thus leading to changes of the pressure distribution at the micro-scale. What is more, due to repetitive movement of the tool at the surface, the roughness characteristics change due to wear. The derivation of contact equations for rough surfaces thus involves mathematical description of the surface geometry by statistical, fractal or other models ([Sextro, 2002; Wriggers, 2002]). Nevertheless, due to its complexity and limited importance in the estimation of contact forces for a haptic tool-tissue interaction, micro-mechanical phenomena will not be further considered.
The development of haptic rendering algorithms for a 6-DOF tool contact with elastic bodies involves an understanding of three main components: the modelling of deformations of the soft body, contact force computation and haptic rendering techniques. In this chapter we will introduce the reader to the related work on these subjects.

3.1 Deformation models of a compliant body

For rigid bodies, the object’s configuration can be described by six variables in Euclidean three-dimensional space. In contrast to this, for compliant bodies infinite number degrees of freedom should be considered. A typical approach to obtain solutions for arbitrary geometries is to discretise a body into a finite number of elements for which the equations of motion are defined. The point of interest of this section are the computation models describing the inner forces of the deformable body and relating the applied external forces to the final shape of the deformed object (see Equations 2.8 and 2.9). An overview of deformable models, with emphasis on surgical simulators and haptic applications, will be given below. For a more comprehensive study on deformable models one is referred to Gibson and Mirtich [1997] and Nealen et al. [2005].

In the following we will describe several existing models, categorizing them to three main classes:

- **geometry based models**: modelling the deformations by simple modifications of the objects geometry;

- **physically based analogies**: describing the deformations as a physical interaction between simple discrete elements (e.g. atoms or cells);

- **continuum mechanics based models**: considering the object’s material as a continuum and applying the constitutive equations over the entire volume.
3.1.1 Geometry based models

First applications which modelled the behaviour of soft bodies were based on geometrical modifications of the surface mesh, i.e. by repositioning the vertices of the elastic object in a visually plausible manner. In this sense, Baur et al. [1998] displaced the surface nodes of a flexible body by a predefined 3D profile functions (Figure 3.1). Starting from the point of contact, the deformation was propagated to the neighbouring vertices according to a chosen profile function. To obtain a realistic visual as well as force feedback, the profile functions have been tuned under the supervision of expert surgeons.

![Figure 3.1](image.png)

**Figure 3.1:** Real-time deformation of an organ as modelled by a predefined profile function in the VIRGY project. Picture taken from [Baur et al., 1998].

Similarly, Basdogan et al. [1998] used second order polynomial functions, fitted to empirical data, to translate the vertices of organs in the vicinity of the contact point along the direction of the virtual tool (Figure 3.2). Modelling of global deformations, was achieved by adapting free-form deformations to spline-based surfaces. With this technique the space enclosing the object is deformed. Applying this concept to spline-based surfaces requires displacement of the control points of the spline surface. Since the user interacts with the model at an arbitrary surface point, the translations of the control points have to be obtained from the displacements of the contact points. In [Basdogan et al., 1998] the method of Hsu et al. [1992] has been applied, which uses a least-squares approximation to obtain the control points for given contact points displacements. The force sent to the haptic device was consecutively computed according to Hooke’s law for a net of springs. Massless virtual springs were assumed between the current position of the contacted vertex and the original positions of this vertex as well as the neighbouring vertices.
3.1. Deformation models of a compliant body

Figure 3.2: Simulation of local tissue deformation by displacing vertices within a radius of influence according to predefined polynomial function. Picture taken from [Basdogan et al., 1998].

Although providing simple implementation and fast real-time modelling of deformations, the geometry–based methods only roughly approximate physical behaviour, which makes them less attractive for the use in surgical simulators.

3.1.2 Physically based analogies

Chain-Mail like algorithms

Deformations of volumetric objects, modelled by the motion of linked elements similar to chain, was introduced by Gibson [1997] (Figure 3.3). In the Chain-Mail algorithm, volumetric elements are linked to their nearest neighbours. The movement of a single element within a certain limit does not affect the neighbouring links in the structure. Links, which are stretched or compressed to their limits, drag or push their neighbours in the desired direction, propagating the deformation through the volume.

One of the disadvantages of this approach is the dependence of the resulting deformation shape on the sequence of the applied forces. To avoid this, Park et al. [2002] proposed to always compute the deformations from the rest shape of the object. This allows to retain the rest shape of the modelled object, if the applied displacement is reversed. This extended approach is called the shape-retaining 3D
Chain-Mail (S-chain mail) model. To compute the deformation force for haptic rendering the stacks between the chain links are considered as springs. The resulting force feedback is thus given by the sum of forces generated by springs involved in the deformation. The S-chain mail model was applied within haptic simulations modelling area contact [Kim, Park and Kwon, 2003; Kim et al., 2005] and a palpation simulator providing cutaneous force feedback via a tactile display [Park et al., 2005; Kyung et al., 2006].

Ideas similar to the 3D Chain-Mail model were also applied by Suzuki and Suzuki [2003] who describe the elastic bodies as clouds of spheres. This, so called Sphere-Filled model, is defined by a one-to-one correspondence to a voxel-based datasets of the organs with explicitly set connections between the neighbouring spheres. Similar as in the Chain-Mail model, a displacement of one sphere is propagated via the neighbours to the whole model. However, in contrast to the Chain-Mail approach, where a single element can move within certain limits without affecting the displacement of the other elements, in the Sphere-Filled model each displacement of a sphere is propagated to the neighbours. This deformation process stops when all overlaps and gaps between the neighbours are resolved (Figure 3.4). This way, the volume preservation of the modelled object is ensured. In the Sphere-Filled model, the reaction force, rendered on a haptic device, is generated proportionally to the average sphere displacement from the rest state.

Particle systems

Particle systems are models composed of a set of points, called particles, with no explicit connectivity defined between them. Each particle is associated with a physical quantity such as position, velocity, mass, pressure etc. The dynamics of the system is described according to Newtonian point mechanics, where particles interact with each other via long term attraction and short term repulsion forces, similarly to the interaction of atoms in liquids.
3.1. Deformation models of a compliant body

![Image of deformation models]

**Figure 3.4:** The idea of propagating the displacement through the neighbouring spheres in a breadth-first search manner. The spheres are marked by a depth level, defined from the sphere in contact.

Mass-Spring system

Another physically based technique, widely and effectively used in real-time computer simulations of deformable bodies, is the Mass-Spring system (MSS). It consists of point-masses (nodes) connected according to an explicitly defined topology. The interaction between the linked nodes is modelled via springs and dampers. In Figure 3.5 an example of a MSS can be seen.

![Image of Mass-Spring system]

**Figure 3.5:** Mass-Spring system showing nodes of mass $m$ connected by springs of stiffness $k$. The topology is obtained according to a hexahedral decomposition of the body. Picture taken from [Gibson and Mirtich, 1997].

Besides springs, defined between two nodes, other relations among a number of mesh entities can be considered. This way forces which compensate for orientation change (e.g. torsional springs as described by Jeong and Lee [2004]), area change, volume change or other arbitrary constraints [Teschner et al., 2004] can be modelled. Adding such force terms results in a more realistic behaviour of the deformed object, especially when subject to large deformations and rotations.

Combinations of other deformation models with MSS have also been presented. Jaillet et al. [1998] used a particle system to model organs, which undergo large
deformations while preserving constant volume. In addition, a MSS was used for modelling the skin of the organ, thus enclosing the particle model (Figure 3.6).

Figure 3.6: Compound deformable model, using MSS for the organ skin and a particle system for the inner part of the organ ensuring volume preservation. Picture taken from [Jaillet et al., 1998].

In the work of Jansson and Vergeest [2002] a mix between a particle system and a MSS, the Molecular Model, has been derived. Mass points are, in fact, spherical regions called molecules, and elastic forces are established by spring-like connections. These are determined on-line considering the distance between nearest molecules of the same as well as of two different objects in the scene. An application of this model within a haptic scenario was presented by Maciel et al. [2004] (see Figure 3.7). Here, a penetration based force feedback was applied between a simple model of a hand and the flexible body. A linear spring considered between the spheres of the hand model produce repulsion force proportional to the penetration depth.

Figure 3.7: User’s hand interacting with soft bodies providing haptic feedback via the Cybergrasp device. Pictures taken from [Maciel et al., 2004].

To further accelerate the computation, e.g. when modelling complex shapes, simplifications of the MSS structure were proposed. Skeleton based objects have been used by Conti et al. [2003]. Here, the volumetric model is composed of elastically
linked spheres, set along the **Medial Axis**, which simulate the deformations of the elastic body. The deformations of the surface are visualised by a separate surface model, which is connected to the skeleton via springs. Similarly, Corso et al. [2002] are considering volumetric objects filled with spheres, centred at the Medial Axis skeleton, with a spline based approximation of the surface. France et al. [2005] are using spheres along a deformable axis with implicit skinning technique to model deformable tubular structures like the small intestine (Figure 3.8). The main advantage of these simplifications is the reduction of the collision detection complexity, since only collisions between spheres or cylinders on a coarse level are considered.

![Figure 3.8: Modelling of a small intestine by a set of spheres placed along a deformable axis. Pictures taken from [France et al., 2005].](image)

**Long element method**

Searching for a highly responsive and reasonably accurate simulation model, for objects composed of elastic skin surface filled with incompressible fluid, Mendoza et al. [2002b] suggested a static Long element method (LEM). In the LEM the volume of an object is decomposed into long beams (long elements) for each considered direction, i.e. the axis of the virtual space (Figure 3.9). While each element is assumed to be filled with fluid, it also obeys Hooke’s Law in the axial direction. Applying Pascal’s principle and the volume conservation law as the boundary conditions, relation between the pressure change (due to contact) and the object’s shape is formulated. To model surface tension, the neighbouring elements are connected via a net of springs at the surface. Using the formalism of LEM, the object can be represented by bulk variables such as pressure, density and volume, which can be easily identified.

The main drawback of the LEM is its validity only for small deformations. For large deformations, the discretisation of the body leads to inconsistent results and absence of volume conservation. Solving this problem by discretisation of the body at each iteration step would diminish the interactivity of the interaction. Lately, Sundaraj et al. [2003] presented as improvement of the LEM, the **Volume Distribution Method** (VDM), which does not require discretisation of the interior of the object. The VDM
3. State of the art

Figure 3.9: A long element with its neighbours (left) and a decomposition of a liver into long elements for 1D (right). Pictures taken from [Mendoza et al., 2002b].

method proved to be able to simulate linear, nonlinear and anisotropic behaviour of flexible bodies.

3.1.3 Continuum mechanics based models

Smoothed particles

This method is an extension of the discrete particle systems (see Section 3.1.2) towards a continuum mechanics. Using the principle of Smoothed Particle Hydrodynamics, approximations of values and derivatives of continuous physical quantities are modelled in a stable and accurate manner. In [Desbrun and Cani, 1996], for example, the discrete values, carried by the particles, are smeared out over the volume by smoothing kernels, thus providing a discrete approximation of continuous functions. Applying these principles to the constitutive equations of elastic materials, a representation of the deformations of organs can be obtained [Hieber and Koumoutsakos, 2005; Hieber et al., 2004]. Particle systems simplify modelling of bodies with complex topologies, undergoing large deformations and topological changes (Figure 3.10).

Finite element method

The Finite element method (FEM) is based on the discretisation of the simulated body into a finite number of elements - providing a mesh structure. The deformation within each element, i.e. the displacement field, is then described by interpolating the displacements of the element’s nodes. The most common elements and interpolation functions, also called basis functions, are respectively polyhedra (e.g. tetrahedra or hexahedra) and polynomials (Figure 3.11). Regardless of the basis functions, FEM provides a piecewise continuous approximation of the continuous deformation
3.1. Deformation models of a compliant body

**Figure 3.10**: Virtual liver decomposed into regular grid of 3209 particles, and cut performed on the model. Picture taken from [Hieber and Koumoutsakos, 2005].

The FEM is the most common approach to solve problems based on continuum mechanics. It is considered as a standard technique in the industry, and therefore most of the necessary parameters of modelled objects can be found in the literature, or can be obtained by standard measuring techniques. For linearly elastic isotropic materials parameters like Young’s modulus $E$ and Poisson’s ratio $\nu$, or the related Lamé constants $\lambda$ and $\mu$, describe the deformation behaviour of the object.

The FEM is also a wide spread technique used in a number of surgical simulators for modelling the soft tissues. In order to allow for modelling of highly complex objects in real-time, a number of simplifications have been introduced. Cotin et al. [1999] sped up the simulation by using the precomputed response of a linear FEM model and the principle of superposition.

Another approach, considering the deformation computation just for the surface vertices was proposed by Bro-Nielsen [1997]. Using a technique called condensation, a stiffness matrix for the surface nodes is created from the system’s stiffness matrix $K$ in a precalculation step. This is done by splitting $K$ to four blocks, each of which describes a relation between surface nodes (denoted by index $s$) and internal
nodes (index \(i\)). The quasi-static equation describing the deformation of the body, i.e. \(Ku = F\) (see corresponding Equation 2.9), can be thus rewritten:

\[
\begin{pmatrix}
K_{ss} & K_{si} \\
K_{is} & K_{ii}
\end{pmatrix}
\begin{pmatrix}
u_s \\
u_i
\end{pmatrix} =
\begin{pmatrix}
F_s \\
F_i
\end{pmatrix}.
\]

Considering that the stiffness matrix \(K\) is constant for linear elastic FEM, a new linear system is created:

\[
\hat{K}_{ss} u_s = \hat{F}_s,
\]

where

\[
\hat{K}_{ss} = K_{ss} - K_{si} K_{ii}^{-1} K_{is},
\hat{F}_s = F_s - K_{si} K_{ii}^{-1} F_i.
\]

The size of the condensed matrix, i.e. \(\hat{K}_{ss}\), is much smaller in comparison to the original matrix, which allows for faster position update of the surface nodes while keeping the response of the system identical to the one obtained by using the full matrix. In addition, Bro-Nielsen points out that only a small number of contact loads is applied and thus the solution can be obtained just by considering the non-zero values of the force vector.

In [Székely et al., 1998], [Rhomberg et al., 1999] the authors approach the speedup of the FEM computation by decomposing the model mesh into several subdomains solved on a computer cluster with 64 processors. This scalable (with respect to the number of processing units) algorithm includes an explicit time integration for each subdomain and an information exchange strategy for inner force propagation through subdomain boundaries.

Basdogan et al. [2004] reduced the complexity of the FEM computations by applying modal analysis to the FEM model. Only the most significant vibration modes of the organs were used to dynamically simulate the deformations and the interaction forces.

Many adaptations were also proposed to enhance the quality of the FEM solution for large deformations or precise local deformations. To extend the behaviour of a linearised elastic model subjected to large rotational deformations, Müller et al. [2002] extract the rotational part of the deformation for each node and compute the forces with respect to the non-rotated reference frame. Nevertheless, in [Müller and Gross, 2004] the authors note, that a per vertex approach introduces ghost forces, and thus an improvement was made by extracting the rotational part per element.

Increasing the level of detail around a contact point, while minimising unnecessary computations, was proposed by Wu et al. [2001] who extended the nonlinear
FEM to the concept of progressive meshes. The described scheme, called **dynamic progressive meshes**, allows for adaptive remeshing of the model while it deforms.

An adaptive approach, which does not depend on the precalculated sequence of progressive meshes, was proposed by Faraci et al. [2005]. Increased precision of the objects deformation was obtained by refining the mesh elements in the contact area. A number of refinement rules for a tetrahedral mesh was proposed, which preserve the compatibility of adjacent elements in the mesh (i.e. avoid the generation of “T–vertices”). The equations of motion for the new vertices were obtained using mass lumping simplifications and an explicit integration scheme.

Another way to locally increase the resolution of the deformation is to refine the basis functions instead of the elements. In this sense, Grinspun et al. [2002] introduced the **conforming, hierarchical, adaptive refinement methods (CHARMS)** for general finite elements. This approach avoids the incompatibility problems arising when refining the mesh elements. Moreover, it provides a general refinement technique independent on the domain dimension, element type and basis function order.

**Boundary element method**

Boundary element method (BEM) is an alternative to FEM. The constitutive equations are limited only to the surface domain of the object by transforming the integral form of the equations of motion into a surface integral, applying the Green-Gauss theorem. Thus, the problem size is reduced by one dimension, providing a computationally more efficient approach than FEM. However, since the computation is restricted to the boundary, the inside of the deformable body has to be considered homogeneous.

James and Pai [2001] used a linear elastostatic model based on the formalism of the boundary value problem for modelling accurate interactions with flexible bodies. High update rate, suitable for haptic applications (Figure 3.12), was obtained by precomputing solutions given in terms of Green functions, which provide a basis for all deformations of the object.

Although the BEM promises fast and more accurate results than the FEM, its applicability is limited. Because of the need of a fundamental solution to a given problem (or an approximate one), BEM can handle only a limited group of linear problems, whereas FEM is widely used for all linear as well as nonlinear problems. Moreover, BEM often needs to evaluate complex integrals and thus leads to more difficult implementation.
Figure 3.12: Virtual interaction with a kidney modelled by BEM, using the PHANToM haptic device. Picture taken from [James and Pai, 2001].

Elasticity theory method

This method is based on Hooke’s law and the continuous model, where Lamé’s equation is discretised by approximating the Laplacian operator and the divergence of gradient operator by the discrete umbrella operator [Debunne et al., 1999]. Since the developed model ensures the same behaviour at any resolution, the Elasticity theory method (ETM) can be easily adapted for local refinement or simplifications in the spatial and temporal domain (Figure 3.13).

Figure 3.13: Liver deformed by the Elasticity theory method with adaptively refined node resolution around the contact point. Pictures taken from [Debunne et al., 1999].

The authors further extend the idea of discretised operators in [Debunne et al., 2000] using a mixed Finite Volume and Finite Element formulation to derive better estimates for the differential quantities, with guaranteed error bounds.
3.1. Deformation models of a compliant body

Tensor-Mass system

The Tensor-Mass system (TMS), introduced by Cotin et al. [2000], presents a combination of MSS and FEM. It is based on the ideas of MSS, since the mass is distributed into points whose motion is governed by Newton’s second law. However, the computation of the internal forces is obtained via the energy minimisation formalism of FEM, using tetrahedral mesh structure. Symbolically integrating the force through the volume of each tetrahedra, stiffness tensors, dependent on the material parameters, are defined for each node and each edge (see Figure 3.14 left).

Figure 3.14: The data structure representation of the Tensor-Mass system (left) – tensors are stored at each edge and each node. Picture taken from [Cotin et al., 2000]. Deformation of a liver (right), based on linear (dark wireframe) and nonlinear (solid) Tensor-Mass system. The rest shape is shown with light wireframe. Picture taken from [Picinbono, Delingette and Ayache, 2000].

Later, this model was extended to capture nonlinear [Picinbono, Delingette and Ayache, 2000] and anisotropic [Picinbono, Lombardo, Delingette and Ayache, 2000] behaviour of the soft tissues (Figure 3.14 right). Moreover, explicit volume preserving constraints were added to the model in order to compensate for the incompressibility of soft tissues.

Method of finite spheres

The Method of finite spheres (MFS) [De and Bathe, 2000] is a fully meshless method which can be used to solve the constitutive equations of elastic materials. In comparison to FEM, there is no need to decompose the object’s body into volumetric elements. Instead, a set of points is spread in the area of interest. These points represent the centres of spheres, on which the continuous equations are approximated with shape functions similarly to FEM (see Figure 3.15). Instead of using polynomials, MFS applies rational functions, generated using the moving least squares method and weighted by radial functions, to approximate the deformation field.
Since no object mesh has to be defined, the MFS avoids the remeshing burden of FEM, when subject to large deformations or when topological changes (i.e. cuts) have to be considered. Nevertheless, the method is computationally very demanding. Even though just a couple of spheres are necessary for realistic rendering of deformations (in comparison to the number of elements needed for FEM), the computation of the integration part is harder due to the use of rational functions and an increase of interpolation points. Therefore, Kim, De and Srinivasan [2003] suggested to use hybrid modelling scheme for soft tissue simulation. They applied linear BEM to model global deformations, whereas MFS was selected in order to capture the nonlinearity of the local deformations around a contact point. To generate the force output for a haptic device, the reaction forces computed by the local model have been used.

### 3.2 Contact force modelling

After presenting different methods to compute the response of deformable bodies, i.e. methods providing approximate solutions to the Equations 2.1 – 2.4 for known boundary conditions (including the interaction forces), in this section we explore the contact force estimation techniques used in physically based computer guided simulations. Now we will present to the reader several methods generally used for collision response between objects in virtual environments. Specific applications to haptic rendering will be discussed in the next section.

In the following we will focus on four techniques. First we present a method in which the contact forces are computed directly from the deformation models, supposed the shape of the elastic body in the contact area is known. Afterwards, the related work for three techniques used to estimate the collision forces, i.e. the penalty force, the constraint dynamics and the impulse–based paradigms, will be described.
3.2. Contact force modelling

3.2.1 Dirichlet conditions

In the previous section (i.e. Section 3.1) we have presented several methods which provide a numerical approximation of the deformation problem for elastic bodies. All deformation models are describing a relation between two unknowns: the displacement field of the elastic body and the applied forces. The corresponding equations can be presented in a compact form:

\[ \tilde{K}(u_x, \dot{u}_x, \ddot{u}_x) = \tilde{F}, \]  

(3.1)

where the function \( \tilde{K} \) and the generalised force vector \( \tilde{F} \) are given by the Equations 2.8 or 2.9. Defining the displacement field in the contact area, e.g. at the nodes of the deformable model, the interaction forces can be obtained by solving the above system of equations. This approach corresponds to a simplification where Dirichlet’s boundary conditions (see Equation 2.5) are applied in the contact area and the deformation of the elastic body is subsequently solved according to Equations 2.1 – 2.6.

In the case when the soft bodies are deformed by rigid tools, the displacements in the contact area can be estimated from the shape of the tool. Therefore, this method is very popular in surgical simulators where mostly only the contacts between the surgical instruments and the patient’s organs are considered. For example, this approach has been applied in [Gibson et al., 1997; Székely et al., 1998; Bro-Nielsen, 1997; Cotin et al., 1999; Laugier et al., 2001; Cavusoglu, 2000; Suzuki and Suzuki, 2003; Kim, De and Srinivasan, 2003; Basdogan et al., 2004] and many others.

Although adaptations of the deformation models to this method are quite straightforward, computing the contact forces following this approach presents several shortcomings. Firstly, the displacement field at the contact point is defined only heuristically. Problems will therefore arise when the tool penetrates the soft body to large extent. Incorrect detection of collision points, and nonrealistic deformations, leading to stability problems, may be obtained.

Secondly, estimating the shape of the contact area can be difficult for contacts between two deformable bodies, as this depends on the stress distribution inside the specimen. Nevertheless, Kuroda et al. [2002] suggested an approach, where the roles of indenter and the soft specimen are simultaneously switched between the two bodies, in order to obtain deformations of two elastic objects in contact.

At last, without knowing the stress distribution at the contact, it is not possible to correctly define the tangential displacements and thus to model frictional effects.

3.2.2 Penalty method

Penalty based force response was first proposed by Terzopoulos et al. [1987], who pioneered the physically based computer simulations of elastic bodies. In his work,
the collisions between the elastic models and impenetrable bodies were simulated based on a potential field defined around each object. The potential energy was expressed as:

\[ E = c_1 \exp \left( -\frac{f(x)}{c_2} \right) , \tag{3.2} \]

where the constants \( c_1 \) and \( c_2 \) determine the shape of the potential, and the function \( f(\cdot) \) is the object’s inside/outside function. This energy becomes prohibitive if the model penetrates the object, and thus the corresponding contact force penalises possible object intersections. The collision force is given by:

\[ \mathbf{F}_{\text{collision}} = - \left( \frac{\nabla f(x)}{c_2} \exp \left( -\frac{f(x)}{c_2} \right) \cdot \mathbf{n} \right) \mathbf{n} , \tag{3.3} \]

with \( \mathbf{n} \) denoting the unit normal vector at the surface. Nevertheless, to define the energy field, this approach requires an implicit description of the object’s surface (given by the function \( f(\cdot) \)), which may not be easy to obtain for complex shapes.

In general, the penalty force can be determined from a penetration measure, usually represented by the depth or volume of intersection and a direction along which the objects have to be moved to resolve the collision. In Figure 3.16 we show an example of detected collision points and computed separation directions, as proposed by Heidelberger et al. [2004], which can be used for the estimation of the penetration force. Given a vector \( \delta \) as the norm of the penetration, the collision force is often approximated by a spring-like analogy:

\[ \mathbf{F} = -k \delta - b \dot{\delta} , \tag{3.4} \]

where \( k \) and \( b \) are defining the stiffness and damping of the contact force, respectively. Note that the measure \( \delta \) does not necessarily represent a linear function of

---

**Figure 3.16:** Detected penetration depths and separation directions of the colliding points. Picture taken from [Heidelberger et al., 2004].
the penetration depth or volume. The penalty based force response works well for small penetrations. However, in interactive, discrete–time simulations, large penetrations can occur due to the size of the time step. This can result in non–plausible penetration depths and directions.

### 3.2.3 Constraint based methods

To fulfil the nonpenetration condition of unilateral contacts, contact forces are considered as bilateral constraints of object positions, velocities or accelerations. The basic solution of computing the constraint forces, i.e. forces which restrict the motion of bodies in contact, the **Lagrange multipliers** method can be used, as described in [Shabana, 1989]. This method solves a composite equation of motion (see Equation 2.8):  

\[
M \ddot{u}_x + D \dot{u}_x + K(u_x) = F_{\text{ext}} - J_c^T \tilde{\lambda},
\]

where \( \tilde{\lambda} \) are the unknown magnitudes of the generalised constraint forces \((J_c^T \tilde{\lambda})\). The term \( J_c^T \) is the transpose of the constraint’s Jacobian, which is defined according to the detected contacts and the form of the constraints. Based on the nature of the constraints and the type of the simulation, relations between forces and accelerations \( f = ma \) (according to Newton’s 2nd law), impulse forces \( j \) and velocities \( j = mv \) (for kinematic manipulation) or velocity changes and impulse forces \( \Delta v = \frac{1}{m} j \) (for a dynamic simulation with collisions) can be considered [Baraff, 1996].

Given a holonomic constraint \( C(u_x, t) = 0 \) on the positions of the bodies in contact, the constraint Jacobian can be obtained by time differentiation [Metaxas and Terzopoulos, 1993]:  

\[
J_c \dddot{u}_x = \frac{d^2 C(u_x, t)}{dt^2}.
\]

Finally, the solution can be obtained by solving a matrix equation of the form:  

\[
J_c M^{-1} J_c^T \tilde{\lambda} = b,
\]

with \( b \) derived according to Equations 3.5 and 3.6 (see [Metaxas and Terzopoulos, 1993; Baraff, 1996] for details).

Although this method delivers dynamically consistent contact forces, which satisfy the given constraints, its drawback is in imposing equality constraints on the contact points of the collided bodies. This may result in sticky effects on contacts which should be split due to the influence of other collisions. While this approach can work well for frictionless contact problems, where the contact topology changes are smooth, i.e. when the bodies are soft and contact velocities are low (see for example [Otaduy and Gross, 2007]), it hinders the modelling of dry friction, for which the non–smooth slip and stick behaviour is to be simulated. Therefore, this method is
more suitable for modelling stiff contacts in articulated bodies, e.g. modelling of joints.

To account for vanishing contacts, inequality constraints are to be taken into account. Baraff [1989] describes a solution, which enforces the impenetrability constraints by considering the valid accelerations of the contact points. First, to model contacts between the simulated bodies, the author differentiates between two types of contacts: resting contacts and collisions. The resting contacts are characterised by zero relative velocity at the contact points, while the collisions represent impacts of two bodies, which are approaching each other and will bounce away.

To formulate the force equations for the resting contacts, Baraff [1989] investigates the relative distances between the contact points (A and B), i.e. the gap function \( g = (x_B - x_A) \cdot n_A \) (see Section 2.2.1). Considering that the points are in resting contact at the collision time \( t_0 \), i.e. \( g(t_0) = \dot{g}(t_0) = 0 \), the accelerations which avoid any penetration are constrained by \( \ddot{g}(t_0) \geq 0 \). Since the accelerations of the bodies depend linearly on the contact forces, the last equation can be formulated as (see [Baraff, 1997]):

\[
AF - b \geq 0,
\]

(3.8)

for a vector of constraint forces \( F \) and relative accelerations at the contact points \( b \). More constraints are defined by considering only repulsive contact forces and by restricting the collision forces to be zero for vanishing contacts. Summarising all these constraints, the contact problem can be expressed in a form of a linear complementarity problem (LCP):

\[
AF - b \geq 0, \quad F \geq 0, \quad F^T(AF - b) = 0.
\]

(3.9)

Note that this formulation corresponds to the Signorini conditions of unilateral contacts (see Equation 2.11), since the valid accelerations at the contact points, i.e. \( \ddot{g} \geq 0 \), implicate the nonpenetration condition \( g \geq 0 \) stated by Signorini.

To solve the LCP, Baraff presents a number of algorithms, approaching the problem heuristically [Baraff, 1989], using quadratic programming, linear programming methods like Lemke’s algorithm [Baraff, 1993] or Dantzig’s algorithm [Baraff, 1994a], and simple iterative Gauss-Seidel like methods [Baraff, 1993]. For more computational methods on solving LCPs in general, one should consult Cottle et al. [1992].

To resolve the colliding contacts between two bodies, Baraff [1989] mimics the problem of resting contacts. According to the Newton’s impact law, the relative velocity after the collision \( (v_i^+) \) can be expressed in terms of the relative velocity before the impact \( (v_i^-) \): \( v_i^+ = -e v_i^- \), for \( e \) being the coefficient of restitution. Due to simultaneous occurrence of multiple collisions, for each contact point it shall be required:

\[
v_i^+ + e v_i^- \geq 0.
\]

(3.10)
Furthermore, the impulses are restricted to act only in the positive normal direction, and to be zero, when $v^+_i$ actually exceeds $-e v^-_i$. Now, imposing the fact, that the post-collision velocities are linearly dependent on the collision impulses, a constraint system of similar form as the one for resting contacts is obtained:

$$v^+(P) + ev^- \geq 0, \quad P \geq 0, \quad P^T(v^+(P) + ev^-) = 0. \quad (3.11)$$

Here $P$ is the vector of unknown collision impulses, and $v^+(P)$ denotes a linear function of the post-collision velocities. This system represents again a LCP which solution can be found with the methods applied for resting contacts. The impulse based contact constraints with a zero coefficient of restitution are also known as the velocity Signorini contact conditions, which, under certain regularity assumptions, can be considered to be equivalent to the original Signorini conditions [Jean, 1999].

Adding frictional forces to the contact phenomenon, [Baraff, 1993] first dealt only with dynamic friction, using analytical methods for the computation. Later, in [Baraff, 1994a], a more reliable model, compensating for static as well as dynamic friction, was presented. In the algorithm, Coulomb’s friction model was considered, which was linearised in two fixed directions. An extension of the LCP formalism to frictional contacts based on polyhedral approximation of the Coulomb’s friction cone, i.e. n-sided friction pyramids, can be found in [Stewart and Trinkle, 1996]. Nevertheless, in 3D contacts a nonlinear relation between the friction forces and the normal load should be considered (i.e. $\|F_f\| = \sqrt{F_{fx}^2 + F_{fy}^2} \leq \mu F_n$). Lately, solutions of the nonlinear frictional contact problem were addressed by Renouf and Acary [2006] and Duriez et al. [2006]. In both approaches, the frictional contact problem is solved by iterative fixed point methods, where the direction of the friction force is determined at each iteration step.

The application of the constraint based contact force computation was also applied to deformable bodies, for example in [Baraff and Witkin, 1992], where geometric deformations were considered. In general, the adaptation to deformable models requires only slight modifications when obtaining the linear relation between point accelerations and the contact forces. Instead of using Newton’s dynamic equations of motion for rigid bodies, this relation is obtained directly from the deformation model. Note that instead of constraining the point accelerations as was done by Baraff, using a time stepping scheme one can consider the Signorini conditions directly by constraining the positions of the contact points at the next time step. For this, a linear relation between the collision forces and the point positions can be obtained from the deformation model and the time stepping scheme (i.e. numerical integration scheme). Such an approach was for example addressed in Jourdan et al. [1998] and Duriez et al. [2006].
3.2.4 Impulse based methods

If two bodies collide, a discontinuity in their velocities emerges. To model this instantaneous change, and thus to avoid interpenetration, force should be applied over an infinitely small time step. This can be easily modelled by applying impulses to the collided bodies. The impulses change directly the velocity of the objects, thus no further integration is needed.

One of the most noted impulse based methods for handling collisions and contacts with friction is the one proposed by Mirtich and Canny [1995]. It is based on an assumption of single contact at a given time. Nevertheless, multiple contacts can be solved using a propagation method, i.e. considering multiple collisions as a temporal sequence of single collisions.

![Figure 3.17](image)

Figurę 3.17: Two phases of collision: compression and restitution. The change from the first phase to the second occurs at the point of maximum compression, for which the contact velocity in normal direction vanishes. The functions $F(t)$ and $P(t)$ denote the force and the impulse, respectively, delivered during the collision. Picture taken from [Mirtich and Canny, 1995].

The main contribution of the algorithm introduced by Mirtich and Canny [1995] is the computation of the required impulse. Each collision can be split into two phases (Figure 3.17): a compression and a restitution. The border between these phases is a point of maximum compression ($mc$). To obtain the final collision impulse, the
impulse variation has to be integrated over the period of contact. The following differential equation is considered:

\[
\frac{d}{d\gamma} \mathbf{v}_{rel}(\gamma) = \mathbf{A} \frac{d}{d\gamma} \mathbf{P}(\gamma),
\]  

(3.12)

where \( \mathbf{v}_{rel} \) is the relative velocity at the contact point, \( \mathbf{P} \) the impulse function and \( \mathbf{A} \) is the inverse of the effective mass matrix seen at the contact points (see [Mirtich and Canny, 1995] for details). \( \gamma \in \mathcal{R} \) is the integration parameter. Instead of choosing time as the integration parameter \( \gamma \), which would lead to integrating forces, the authors proposed to use the normal component of the applied impulse \( \mathbf{P}_n \). When the collision begins, this scalar is zero. To obtain the final point of integration \( \mathbf{P}_f \), the Poisson hypothesis is applied:

\[
\mathbf{P}_f = (1 + e) \mathbf{P}_{mc},
\]  

(3.13)

for \( \mathbf{P}_{mc} \) denoting the magnitude of the normal component of the impulse imparted up to the point of maximum compression. The parameter \( e \) is the coefficient of restitution \((0 \leq e \leq 1)\). The point of maximum compression is identified by tracking the relative velocity during the integration, which vanishes at that point.

Note that, instead of Equation 3.13 one could use Newton’s kinematical definition of the restitution coefficient, i.e. relating the relative velocity change to the relative velocity before impact \( (\mathbf{v}_{rel}^-) \): \( \Delta \mathbf{v}_{rel} = (1 + e) \mathbf{v}_{rel}^- \). This solution was applied for example by Bender and Schmitt [2006]. Nevertheless, since Newton’s impact law is equivalent to the Poisson hypothesis only for centric impacts without friction, for general frictional contact an overall increase of energy can emerge. In [Bender and Schmitt, 2006] this is avoided by explicitly ensuring that the sum of all impulses applied to a pair of contact points is in positive normal direction.

To further simulate friction, Mirtich and Canny [1995] also integrate the impulses in the tangential space and track the tangential velocities. When the relative tangential velocity drops to zero, it is determined whether the friction is sufficient to maintain sticking using the Coulomb’s law and comparing the tangential impulse. If sticking occurs, the integration is further processed only in the normal direction. Otherwise sliding resumes and the integration continues as before. Tracking the velocities and the applied impulses the transition between sticking and sliding can be correctly simulated.

To resolve resting contacts, an approach considering a number of small, frequent collisions between the bodies, called microcollisions, is used. Each of these microcollisions can be resolved using the presented algorithm, with only small modifications made for correct modelling of static friction under continuous contact (see [Mirtich and Canny, 1995] for details). The authors state, that applying microcollisions leads to correct macroscopic physical behaviour, as verified by experiments.
Overall, the impulse-based computation is conceptually simpler and more robust than the constraint based method described by Baraff [1989]. Sauer et al. [1998] state that the impulse based method is more suitable for non-permanent contact (e.g. bouncing behaviour) while the constraint based technique (considering position constraints) is better for permanent contacts. Nevertheless, the physical reliability of the friction modelling makes both of these methods predestined for the use in virtual reality and engineering applications.

In [Ruspini and Khatib, 1997], it was shown, that the formalism of the impulse based contact dynamics and velocity-based constraint method are equivalent. To solve the complementarity constraint problem, the authors therefore have chosen the approach of Mirtich, i.e. integrating the impulses over the period of contact to obtain the change in relative velocity, instead of applying algorithms which solve the linear complementarity problem as described in [Baraff, 1994b]. Nevertheless, the formalism based on velocity constraints presented a clear view of how multiple contacts are coupled. Stacking contact space Jacobians of each collision point into one operational space mass matrix (the matrix A in Equation 3.12) the impact equations can be solved by handling multiple collisions simultaneously.

### 3.3 Haptic rendering methods

In this section we present an overview of approaches used for haptic rendering of tool contacts with the virtual environment. Our goal is to provide a survey of applications considering different aspects of a tool interaction. In particular we will revisit 3-DOF and 6-DOF force feedback simulations, tool contacts with rigid and deformable bodies, single and multiple points contacts, interaction with simplified tools and with tools of complex geometries, techniques compensating for the shape of the tool in the contact area, and integration of frictional effects. Last but not least, at the end of this section several methods will be discussed which help to bridge the gap in update rates between what can be obtained for the simulation of the virtual bodies and what is necessary for stable haptic feedback.

#### 3.3.1 Contact force models applied to haptic rendering

Before we start introducing the reader to the various techniques used when displaying contact forces, we have to mention two issues which influence the style of haptic rendering. Both of these issues, the limited stiffness\(^1\) and the bounded refresh

\(^1\) We would like to note, that due to availability, in all our experiments we used only impedance controlled haptic devices (e.g. PHANToM by SensAble and Delta by Force Dimension), for which force output has to be defined based on a measured position input. Therefore, we will consider
rate represent hardware limitations of the haptic devices, which affect the mode of computing collision forces.

Due to limited stiffness (which is usually around 10 N for a desktop haptic device), the haptic device cannot effectively stop the user from penetrating into the surface of the virtual body, even for slow motions. Moreover, the bounded refresh rate (usually around 1 kHz) limits the range of the achievable virtual stiffness that a haptic device can render while guaranteeing stability [Colgate and Brown, 1994]. As a result, when modelling interaction between a virtual object controlled by the haptic interface, the so called haptic interface object (HIO), and other bodies in the scene, it has to be considered, that the HIO can always penetrate the other objects.

The fact that we cannot avoid penetrations of the HIO with the collided bodies suggests that the Dirichlet boundary induced and the penalty based methods (described in Section 3.2) are the most intuitive techniques to be used for haptic rendering.

**Dirichlet boundary**

As already mentioned in Section 3.2.1, computing the contact forces by defining the displacements of the deformable body in the contact region is a popular approach in surgery simulators. Most of these techniques were applied for minimally invasive surgery simulators, where the interaction was modelled point based, representing only the tool-tip of the surgical instruments [Delp et al., 1997; Kühnapfel et al., 2000; De et al., 2002]. Nevertheless, extensions to multiple contact points are straightforward, i.e. by imposing surface displacements based on the motion of the tool [Székely et al., 1998; Debunne et al., 2001; Bruyns and Montgomery, 2001; Kim, Park and Kwon, 2003; Kim et al., 2005]. To the best of our knowledge, this approach has been always applied considering either frictionless contacts or an infinite friction (e.g. [Delingette et al., 1999]).

The main difficulty of this method is in providing fast and robust simulation of the deformable body, which is necessary in order to guarantee stable force feedback. This is also the reason, why much research effort was devoted to the development of real-time methods for simulating the physical behaviour of deformable tissues. A brief overview of different deformable models as well as several speedup techniques has been already given in Section 3.1.

The advantage of this method is that for steady contacts, slow motions and contact forces in the output range of the haptic interface the deformed body can be visually only impedance control of the haptic interfaces throughout the whole document. However, this should not be seen as a limitation of our haptic rendering approaches. Techniques exist, e.g. virtual coupling [Adams and Hannaford, 1998], which can effectively turn the impedance based approach to an admittance based one and vice versa.
displayed without exhibiting any interpenetration with the HIO. This is especially valuable for applications using collocation setups and other augmented reality systems. Nevertheless, for large penetration (which can not be avoided considering the hardware limitations of the haptic device) the tissue model would have to be pushed or pulled farther than is realistically possible. To avoid this, the virtual instrument should be decoupled from the input device [Delp et al., 1997], loosing thus the feature of realistic contact appearance.

Penalty force

The penalty based force response is typically used for rigid body interactions, where the direction of the penalty force can be effectively precomputed [McNeely et al., 1999; Kim et al., 2002]. For deformable bodies, the process of obtaining a plausible and consistent penalty measure is more difficult due to the fact that the geometry of the body changes. Therefore, the penalty method has been applied to simple models, limited to small penetrations. For example, in [Meseure et al., 2003; Maciel et al., 2004] the penalty forces were computed between objects represented by a set of spheres. The use of a simplified model allows for efficient force computation, fast enough for haptic rendering.

Considering that for discretised elastic models the point load eventually leads to singular displacement of the node in contact if the mesh is refined in the contact area, a point–based contact is an ill-posed assumption and care must be taken to meaningfully define the interaction. In reality, all contacts happen over a small area. To address this issue, James and Pai [2001] suggest to distribute the load over using predefined shape masks. Spherically symmetric and radially decreasing functionals have been chosen to construct the vertex pressure masks on smooth surfaces.

The conventional techniques for penalty methods are based on point contact, using only the features (e.g. vertices, planes, etc.) which penetrate into the other body for the force generation. This leads to non–continuous force changes, when contact points appear or disappear. To avoid this, Hasegawa and Sato [2004] use distributed contact points on the whole surface, and integrate the penalty forces over the whole contact area.

Another problem emerges when setting up the stiffness coefficient of the penalty force. If a constant value is used, big differences in the contact stiffness will be felt by the user depending on whether only a very localised contact (i.e. point like contact) or multiple intersections, respectively large contact areas, have been detected. Such an effect can be observed when comparing an indentation with the tool tip and with the whole shaft of a tool. Allowing the net stiffness to become very large would lead to instabilities of the force feedback associated with fixed time-step numerical integration. To avoid this problem, McNeely et al. [1999] limits the
maximum total stiffness of the penalty force. To also compensate for the irregularity of the stiffness distribution, due to the spatial arrangement of the contacts, Kim et al. [2002] cluster the contacts based on their proximity and distribution in the 3D space. Note that contact clustering also effectively reduces the number of contacts which define the force output, speeding up the contact force computation for objects of complex geometries.

Applying the penalty forces only when the object is in contact ignores the dynamics’ momentum of the colliding object, which can induce deep penetrations, thus reducing the stability and also the realistic appearance of the simulation. McNeely et al. [1999] proposed to use a pre-contact breaking force which decreases the velocity of the object approaching an obstacle. More specifically, they apply a force which dissipates the kinetic energy of the object within one time step.

An adaptation of the pre-contact breaking approach has been implemented by Johnson and Willemsen [2003]. For haptic rendering of contacts between rigid bodies they suggest to use forces which are preventing the models from colliding as they approach each other, rather than moving them apart once interpenetration occurs. For this purpose, local minimum distances are computed within a cutoff distance between the pair of objects for which spring-like forces are applied to avoid collisions. Besides maintaining real-world constraints by applying pre-contact forces, the authors also claim, that minimum distances can be computed faster than the penetration depths.

The penalty method is in general not designed for frictional contacts. Since the penetration volume is computed to minimise the distance between the feature sets at the surfaces of the two bodies, only normal force components are estimated. Nevertheless, it is straightforward to apply viscous friction by defining forces which impede sliding motions along the object’s surface. To include dry friction, modelling stick-slip phenomena, an approach presented in [Hayward and Armstrong, 2000] can be applied. Here, the contacts are tracked in the time domain and an adhesion point is defined to which the frictional forces are computed (see applications in [Kim et al., 2002; Hasegawa et al., 2003]). Using the distributed spring model of Hasegawa and Sato [2004] with adhesion points defined for each contact, it is possible to precisely calculate the friction forces and also the friction torques.

God–object and the virtual proxy

The main difficulty when applying the Dirichlet boundary and the penalty based paradigms is in determining the points at the surface of the deformable body to which the displacements or the forces will be applied. If no precautions are taken, for deep penetrations this can lead to singularities due to nonexistence of a unique
solution, discontinuities of contact forces if the moving probe gets suddenly attracted to another nearest surface part and possible passing through small and thin objects.

To address these problems Zilles and Salisbury [1995] proposed the god-object (GO) method. The GO is a physically simulated virtual object, attached to and controlled by the haptic device, which is explicitly constraint not to penetrate into other bodies. As such, the GO represents an ideal position of the HIO, supposed the haptic device and the object are infinitely stiff. The position of the GO is defined as an optimisation problem. It is located at a minimum distance from the HIO, while its interframe trajectory is constrained by the surface of the indented body. Zilles and Salisbury [1995] solve for the position of a point-represented GO using Lagrange multipliers, restricting the GO to a set of active constraints.

Ruspini et al. [1997] extended this approach, simulating effects like force smoothing (i.e. force shading), friction, compliant contacts and textures simply by changing the position of the GO. This paradigm is known as virtual proxy (VP)\(^2\). The authors also point out, that the GO should be seen as a physical body, to which geometric as well as physical properties can be assigned (i.e. shape and mass).

[Ho et al., 1999] also implemented the ideas of point-based haptic rendering using a GO, referring to the ideal haptic interface point. To obtain the constrained position of the GO they locally update the GO to the nearest position on the surface primitive (i.e. vertex, edge or face) in the one-ring neighbourhood of the current GO’s position.

Niemeyer and Mitra [2004] introduce dynamic proxies which enable greater control over the behaviour of the proxy object in force based dynamic simulations. Zero–order (i.e. quasi-static), massless first–order and second–order dynamics have been presented by the authors.

Assuming that we have obtained the position of the GO, to compute the force which should be rendered on the haptic device, simple impedance control techniques can be used [Zilles and Salisbury, 1995; Ruspini et al., 1997]. Imagining a virtual spring and damper connecting the HIO to the GO, the discrepancy in positions and velocities between these two objects will generate forces which will drag the HIP towards the surface. On the other hand, the stretched spring will apply forces to the GO (admittance control) which will pull the GO towards the HIP position, thus trying to minimise the distance between these objects. Applying this sort of bilateral control is known as virtual coupling (see also Section 3.3.1).

To apply static friction, i.e. sticking of the HIO to the contact point, it is sufficient to fix the current GO’s position at the contact polygon [Ruspini et al., 1997]. The friction force will thus be modelled by the tangential component of the force given by the spring connection between the GO and the HIO. Whether stiction should be

\(^2\)Note that the GO and the VP are used congruently throughout the literature for modelling the same behaviour. Referring to either of these models is thus a way of personal preference.
applied can be simply checked by comparing the tangential and the normal force components of the estimated contact force, subject to Coulomb's law (i.e. $\|F_t\| < \mu_s\|F_n\|$). Dynamic friction can be modelled by bounding the amount that the GO can travel in one time step [Ruspini et al., 1997], or by setting the new position of the GO to a location on the line between the old GO and the predicted GO position (nearest to the HIO) where the tangent force component of the spring connection will be equal to the dynamic friction [Ho et al., 1999]. The latter one is also known as the friction cone algorithm. The application of the friction cone has been mostly promoted by Harwin and Melder [2002] who simulated multi-finger manipulation of objects using multiple point-based GOs.

The main process in the implementation of the GO paradigm is to find the constraint position of the GO. While it can be considered as a simple problem for a single point interaction, in general an analytical method is to be applied for the motion simulation of the GO (i.e. the constraint or the impulse based paradigms; see Sections 3.2.3 and 3.2.4). This, however, leads to computationally demanding problems when multiple contact points are considered (NP-hard problems as stated by [Baraff, 1994]).

Nevertheless, several heuristical methods have been presented for simple shaped tools. A simple extension of the point-based GO to a line-segment interaction was described by Bielser [2003]. If a two-point intersection of the line-segment was detected, an interaction point was defined as the mid-point of the intersected part. A point based VP was then simulated for the mid-point. This model did not allow for multiple collisions of the line with the surrounding bodies and thus was limited to 3-DOF force feedback.

Basdogan et al. [1997];[Basdogan et al., 2000] developed ray-based rendering, for which the tools can be modelled by connected line segments. To refine the GO position (i.e. the surface representation of the line segment), different contact types of the line segment with the vertices, edges and polygons are analysed. The GO is translated along the surface, corresponding to the current contact type, until a new position is found. Considering the discrepancy in position as well as orientation between the GO and the HIO, forces and torques can be reflected to the user. This solution is suitable for MIS simulators, where most of the instruments can be approximated as simple line-objects [Basdogan et al., 2001, 2004].

In [Laycock and Day, 2005] a similar, rule-based approach was used for haptic rendering of contacts between 3D polygonal objects in a rigid scene. All transitions between the surface features (vertices, edges and faces) of the two polygonal objects in contact have been tracked, in order to define a new valid position for the GO. The defined rules ensure, that the tracked features are the closest ones at any given time. The authors also presented this approach for modelling contacts of a deformable, beam-like tool with convex objects. Nevertheless, this approach was designed for
tracking of a single contact at a time. A collision detection query was run at each
GO update step in order to detect whether a new contact point has to be set.

A more complex decomposition of the contact problem has been presented by Luo
and Xiao [2005]. For a virtual assembly task of inserting a deformable tube into a
rigid part, all valid contact states have been organised into a contact state diagram
according to which appropriate contact transitions could be tracked. The contact
force has been defined for each contact state, taking into account the deformation
and friction effects.

Extensions to a general 6-DOF GO interaction have been presented only recently.
Simulating frictionless contacts between rigid bodies of arbitrary geometry has been
presented by Ortega et al. [2006]. The authors apply Gauss’ principle to project
the generalised accelerations (linear and angular) of the coupled virtual object (the
GO) to the set of feasible accelerations defined by the contact constraints. For
rendering the contact forces, a novel method, the constraint based coupling (see
Section 3.3.1), has been proposed.

Duriez et al. [2006] model frictional contacts between deformable bodies. The au-
thors consider Signorini’s law and the nonlinear Coulomb friction law (modelling the
frictional phenomena by the full friction cone) to be applied at the contact points,
leading to a complex nonlinear complementarity problem (NCP). To solve the NCP,
a Gauss-Seidel like algorithm has been implemented. This algorithm computes the
contact forces, according to which a quasi-static simulation of the GO is processed.

In [Otaduy and Gross, 2007] the authors focus on modelling contacts between rigid
tools and deformable bodies. They define frictionless velocity based constraints, and
solve for the dynamic motion of the GO using the method of Lagrange multipliers.
Although this approach leads to sticky contacts, for compliant environments and
relatively slow movements it presents an acceptable tradeoff between speed and ac-
curacy. Nevertheless, the main contribution is in applying a novel linearised contact
model, dynamically consistent with the constraints, for the fast haptic update.

**Virtual coupling**

Virtual coupling (VC), introduced by Colgate et al. [1995] and researched in more
detail by Adams and Hannaford [1998], is a technique, which models a connection
between a haptic interface and any virtual object (VO) set in the simulation en-
vironment. Although the VC method can be seen as a motion control mechanism
for the VO, the primary goal in using VC is to restrict the impedances modelled
in the simulation world to those, which can be stably rendered with the haptic de-
vice. Thus, instead of sending the collision forces from the simulation directly to
the haptic output, the forces are adapted to the mechanical properties of the haptic
interface, i.e. to the dynamic range of achievable impedances (so called “Z-Width”
3.3. Haptic rendering methods

This approach is especially suitable for haptic rendering of infinitely stiff contacts between the GO (or the VP) and the virtual environment. However, the use of VC is not restricted to these constraint based models. In [McNeely et al., 1999; Otaduy and Lin, 2006], for example, it was applied to penalty based contact force models. The central notion of the VC paradigm is the dynamic simulation of the motion of the VO controlled by the haptic interface. More details on the main ideas, issues and implementation of the VC technique can be found in Section 5.2.

Considering linear and angular springs between the VO and the HIO is the easiest and the most common model of VC, applied for example in [McNeely et al., 1999; Duriez et al., 2006; Otaduy and Gross, 2007]. Nevertheless, to cope with hardware limitations like haptic device saturation, nonlinear springs have been suggested [Wan and McNeely, 2003; Otaduy and Lin, 2006]. Niemeyer and Mitra [2004] also suggested to add an acceleration dependent term which would enhance the perception of impacts. Finally, the method of VC, applied to articulated bodies with more DOF than available on a haptic interface was described by Meseure et al. [2004].

To enhance the stability of the force computation, especially for large contact stiffness, implicit integration schemes should be applied for the dynamic motion simulation of the VO [Otaduy and Lin, 2006]. For haptic simulations, particularly, this approach increases the transparency of the tool manipulation. Mainly, the interactions with tools of little mass can be stably modelled, and thus the perception of inertial forces, felt by the user when moving in free space or when sliding over a surface, is greatly reduced.

A similar technique to the virtual coupling, but adapted for the constrained motion of the GO was proposed by Ortega et al. [2006]. In their method, named constrained coupling, the force provided for the haptic output is computed only from the constrained component of the coupling force. Therefore, while during collisions the true contact forces are rendered (scaled by the coupling stiffness to guarantee stable interaction), for free space motion no force is felt by the user (Figure 3.18).

3.3.2 Multirate architecture

A common problem in haptic rendering is that the simulation of the objects in the virtual environment is too slow to be used for haptic rendering. The main bottlenecks are a robust and fast collision detection algorithm, the computationally expensive simulation of the deformations of the elastic bodies, the formulation of the contact constraints and the calculation of the contact forces. To bridge the difference between the update rates obtained from the virtual world simulation and sampling rates necessary for stable haptic feedback, a multirate architecture of the application is usually devised. In a slow process, an accurate interaction between
the tool and the environment is computed. At the same time, an approximate and simple intermediate representation of the contact is updated. In a fast process, the contact forces are synthesised, based on the intermediate representation, and sent to the haptic device for output.

To smooth the force output, Zhuang and Canny [2000] linearly interpolated between the last two positions of the virtual proxy given by the simulation thread. Picinbono, Lombardo, Delingette and Ayache [2000] suggested to extrapolate the contact force in the space domain and Maciel et al. [2004] applied a low pass filtering of the force signal coming from the simulation thread. It has to be mentioned that the virtual coupling paradigm, applied in numerous applications, also represents a low pass filtering approach. Although these techniques cause a slight delay, acceptable results were obtained in most of the applications by making the forces vary smoothly.

An interaction with simple models, locally approximating the geometry at the contact (by planes, spheres, etc.), is another popular choice. Bielser [2003] models a simple frictional interaction with a tangent plane in the haptic loop. The tangent plane is updated by the simulation loop, taking into account the actual curvature of the surface at the contact point. Mendoza and Laugier [2001] use several facets around the contact point to define a low complexity model to which a constraint
based GO simulation is applied in the haptic loop. The corresponding faces are determined by the collision detection algorithm running at a low priority.

A common approach to enhancing the stability of the force feedback, when interacting with elastic bodies, is to define a low complexity deformable model which locally approximates the behaviour of the elastic object. Astley and Hayward [1998] implemented a multiscale finite element model, where a coarse linear FEM was used to predict the overall behaviour and a fine scale FEM, run at the haptic rate, was applied locally around the actual interaction point. Similarly, Cavusoglu [2000] explored model linearisation and reduction techniques to obtain local low order approximations of the deformable model with adequate frequency response. Mendoza et al. [2002] also construct a separate deformation model, covering only some elements around the contact. In the simulation loop the inverted stiffness matrix of the buffer model is obtained. The forces corresponding to contact induced displacements can then be evaluated at the fast refresh rate.

Pre-calculations speeding up deformation simulation have also been exploited. Cotin et al. [1999] and James and Pai [2001] pre-computed nodal stiffness from a linear system which directly relates the nodal displacements to forces. The deformation response of the elastic object was then calculated by the principle of superposition. Although this method efficiently reduces the computation time for the deformations, it is applicable only to linear elastic models. To overcome this limitation, Mahvash and Hayward [2003] suggest to pre-record the force response over the whole surface of the elastic body. The contact force is then synthesised from the response curves stored at the nodes of the mesh.

The techniques mentioned in the previous paragraph can only be used to speed up the deformation calculations, and are therefore ineffective for cases where the collision detection, the formulation of the contact constraints or the computation of the contact forces present the main realtime limitation. These issues apply especially in conjunction with the constraint based simulation of the virtual objects. In [Otaduy and Gross, 2007] the formulation of the constraints and the computation of the contact forces presented the main bottleneck. For fast update of the force, dynamically consistent inverse contact Jacobians are formed, while solving the constraint based problem at the slow rate, and used within a linearised contact model. Moreover, the authors simulate two separate virtual bodies, coupled to the haptic interface. One of them is used in the simulation loop for detecting collisions and formulating the constraint contact problem. The other one is used in the haptic loop to convey the forces computed by the linearised contact model to the user.

To reduce potential discontinuities in the constraint based force output, obtained due to delayed detection of constraints, the new constraints can be added gradually to the simulation, as suggested by [Ortega et al., 2006]. Nevertheless, this approach

\(^3\)The authors reported being able to use around 50 elements.
assumes that it is possible to solve the constraint based equations within the fast haptic loop.
System overview

One of our aims is the development of a simulator for haptic interaction with virtual objects, which will be used as a testbed for the later presented haptic rendering procedures. The basic building blocks of such a physically based simulator include:

- **object modelling**: providing a 3D representation of the virtual objects for the physical modelling, collision detection as well as visualisation;

- **physical engine**: controlling the dynamics of rigid as well as deformable bodies in the scene; it includes the application of a selected deformation model together with its parameter setting and a numerical solver;

- **collision detection**: checking for intersections of two given objects and possibly also for self intersections of the deformable body;

- **collision response**: determining the interaction forces at the contact points and applying them to the collided bodies;

- **haptic rendering**: displaying the net forces via the haptic device to the user.

In the following we will describe the architecture of our simulator and present the details of the main system components dealing with the modelling of the deformations of the elastic body, the collision detection of the deformable body with rigid tools, as well as the distribution of the contact forces to the soft body. The contact force computation and the haptic rendering modules will be described in the next chapter. To clarify the design of our simulator and to introduce areas where the main implementation emphasis will be put on, we will begin with a discussion of several implementation issues.
4.1 Issues of a successful haptic simulator

The main difference when providing visual and haptic feedback in virtual simulation systems is that of fast update rate. While refresh rates of 20 Hz are sufficient for the visual presentation of a smooth motion, for haptic rendering update rates of several hundreds Hertz are required. This is mainly due to the sensitivity of the humans tactile system, which notices vibrations with a frequency up to 1 kHz, with the highest sensitivity around 250 Hz where submicron amplitudes are detectable [Srinivasan and Basdogan, 1997]. Nevertheless, the refresh rate necessary for a stable haptic feedback depends on the contact stiffness which is to be reproduced. While for the interaction with soft bodies update rates of 250 Hz may be sufficient, for a stable haptic rendering of stiff objects usually update frequencies of 1 kHz are required. However, even for these high frequencies the transient signals as are felt for example when tapping on wood or metal are not reproducible [Kuchenbecker et al., 2005].

In [Colgate et al., 1995; Adams and Hannaford, 1998] it was shown that the stability of a force-feedback system can be ensured by providing a passive virtual environment. In particular, the passivity conditions for haptic rendering of virtual walls were formulated in [Colgate and Brown, 1994]. It follows that a sufficient condition for passivity can be expressed in terms of the stiffness $K$ and damping $B$ of the virtual wall, the inherent damping of the haptic device $b$ and the sampling period $T$:

$$b > \frac{K T}{2} + |B|.$$ 

As a conclusion, one possibility for enhancing the stability of the interaction is to increase the physical damping of the device. This, however, would lead to reduced transparency of the force feedback, in that the inherent dynamics of the haptic interface would be felt by the user instead of the dynamics of the virtual environment which we wish to display. Therefore, in order to render stiff contacts, the sampling rate should be maximised. This result also shows that the update rate can be reduced for stable rendering of soft bodies, depending on the compliance of the elastic object.

To compensate for the low update rate of the physical simulation, i.e. soft body deformations, multirate approximation techniques have been applied (see Section 3.3.2 for examples). The main idea is the use of simple, local interaction models for the fast computation of the contact forces rendered at the haptic device. While the local model is designed to run at refresh rates up to 1 kHz, it is updated from the simulation environment, for which the refresh frequency is lower, limited mainly by the collision detection system and the deformation model. An important issue at this point is to guarantee a smooth update of the data describing the local model, since discontinuities would destabilise the haptic interaction. This can be often managed
by filtering or predicting (e.g. extrapolating) the input data given by the simulation system. Finally, the slow refresh rate of the virtual environment can introduce delays in the perception of impacts, thus compromising the haptic sensation of the virtual world. Therefore, a reasonable tradeoff between the update speed of the simulation system and the strength of the applied approximation technique has to be found for each specific haptic simulation.

To conclude, the main issues in the design of a haptic simulator are the fast update rate of the simulation system, thus maximising the performance of the collision detection and the deformation model; and the application of simplified contact force models, running at refresh rates of 1 kHz, which stabilise the interaction and improve the continuity of the haptic output.

4.2 System design

The task of virtually touching deformable objects is very challenging from the computational point of view. It includes issues of complex deformation calculations as well as the need of fast contact force generation. To guarantee haptic refresh rates of 1kHz, we split the application into three major threads. The fastest thread, running with real-time priorities, is used for the haptic loop, the second thread is running the simulation loop and the third thread is kept for the visualisation and user interaction via mouse or keyboard. While all the data are managed by the simulation loop, the other two threads are accessing the required data asynchronously. This way, all threads work with the most recent state of the required information.

![Figure 4.1: The decomposition of the main application into three asynchronous threads.](image)

The simulation loop is the computationally most expensive part of the whole system as it is processing the overall object behaviour in the virtual environment. It utilises the physical simulation of the deformable body, as well as the collision detection between the objects in the scene. All contacts between objects not driven by the haptic device are resolved in this thread, while the interaction with the haptic driven
tools is managed in the haptic loop. The last valid contact forces between the haptic tools and other objects in the scene are accessed by the simulation loop only when necessary. The update rate of the simulation loop is dependent on the complexity of the deformable object, since the deformation calculation uses most of the computational resources. In our simulations we achieved refresh rates from 50 to 200 Hz for objects composed of 600 to 4000 tetrahedra, respectively (see Table 4.2 for details).

The haptic loop is run at 1kHz. On one hand this is considered as a standard for the haptic feedback, on the other hand this value is very often set by the drivers of the haptic device\textsuperscript{1}. Lowering this refresh rate results in reduced stability of the interaction, especially when a collision occurs and larger force discontinuities appear.

Because of the need for a fast update rate, all computations involved in the haptic loop have to be kept simple and tuned to run fast. In this thread we update the tool position driven by the haptic device and also the collision information, in case the tool was previously detected to have collided with the soft body. If collision points exist, the deformation and the frictional forces are computed for each of the contact points. These forces are then sent to be rendered on the haptic device.

At each time instance, the user is in fact interacting with a static representation of the deformable body. Considering that the contact points are defined on a triangular surface mesh, a local model composed only of a set of planes, defined at each contact point, is used to evaluate the contact forces within the haptic thread. When updating the local model according to a new deformation state of the soft body as well as when crossing the edges of the triangular surface, smooth force output has to be guaranteed. We address this issue by employing simple filtering, interpolation and force shading techniques in the haptic thread (more details will be presented in Chapter 5).

The visual thread is run at 30 Hz as this is sufficient for the simulation to appear continuous. However, highly realistic appearance of the scene was not of our intent. Especially, we did not concentrate on a credible visualisation of the deformed soft object. Our main aim was to faithfully recompute the contact forces and display them haptically to the user.

It has to be noted, that a realistic visual feedback can enhance the perception of the haptic interaction. The fact that the visual feedback influences the force sensation has been discussed in several studies. By introducing incongruencies between the visual and the haptic modalities, strong bias towards the visual perception was observed. It has been for example shown by Srinivasan et al. [1996] that the perception of stiffness can be biased by the visual feedback. This result is particularly useful

\textsuperscript{1}The drivers of the PHANToM\textsuperscript{TM} device (by SensAble Technologies Inc.), which we were using throughout our studies, are enforcing an update rate of 1kHz for the haptic loop.
when haptically displaying rigid objects. If the visual representation of a tool driven by a haptic device stops at the surface of the rigid body, the perception of stiff contact is enhanced although the real position given by the haptic device would set the virtual tool inside the rigid object. Even if the influence of the visual feedback on the provided haptic cues was not part of our study, we tend to display all virtual objects in a collision free positions and not at the position defined by the haptic device in order to obtain realistic appearance. This idea particularly corresponds to the haptic rendering paradigm based on a virtual proxy (see Chapter 5) which we will follow.

4.3 The implementation basis

In our virtual environment the simulated objects are described by a tetrahedral mesh. This provides a triangular surface structure, suitable for the collision detection algorithm, as well as a volumetric structure defining the elements of the chosen deformation model. The deformations are simulated by an approximate physical model - the Mass-Spring system (MSS), presented in a dynamic formulation. Real-time interaction is provided for simple point-based rigid tools, which enables modelling of probe-like tools of various shapes (i.e. baton, hook, etc.). In order to fulfil real-time requirements necessary for a stable haptic feedback, we opted for a simple nearest feature tracking approach for detecting collisions between objects in the scene (see Section 4.4.4). The detection of self intersections for the deformable bodies, has not been included in our simulator.

The implementation of the nearest feature tracking presented in this thesis does not allow for general detection of contacts between arbitrary shaped bodies, i.e. concave objects. However, the idea can be extended for such objects by the use of another level of collision detection algorithm running on a coarse scale. Within our simulation scenarios presenting simple shaped objects like cubes, cylinders or spheres, we did not experience crucial problems originating from our choice of the collision detection algorithm. On the contrary, the use of the simple approach allows for efficient collision handling for tools described by a set of predefined points.

4.4 Components

4.4.1 Mass-Spring system (MSS)

Choosing a suitable deformation model for the modelling of soft tissues is a difficult task as biological soft tissues itself are complex materials to represent. They show characteristics such as viscoelasticity, inhomogeneity, anisotropy, or load cycle
conditioning. However, the development of haptic rendering algorithms modelling soft tissue interactions can be considered independent of the internal deformation model of the soft body. The main requirement on the deformation model is high computational effectiveness, since it is to be applied in an interactive environment. Therefore, for our investigations we restricted the choice of the modelled soft materials to simple, linear elastic objects. For these, several deformation models exist which can be effectively computed in real-time (see Section 3.1).

For simplicity and in order to maintain the broadest applicability, we selected a Mass-Spring system (MSS) for our experiments. The MSS delivers fast, physically-based force computation for linear elastic bodies. Moreover, the MSS proved to be useful in several surgical simulators [Kuhn, 1997; Çakmak and Kühnapfel, 2000; Keeve et al., 1999; Tendick et al., 2000; Montgomery et al., 2001; Webster et al., 2002; Bielsa, 2003]. While other deformation models could also have been used, many of them are computationally expensive and some limit the generality of interaction with 3D objects (e.g. through the use of pre-recorded force feedback).

Our MSS is based on a tetrahedral structure of the object we wish to simulate. The vertices of the tetrahedra are the mass points and the edges define springs and dampers. We use linear springs with dampers, also known as Kelvin-Voigt elements, between the nodes of the model.

To define the inner forces of the model we follow Baraff and Witkin [1998]. The forces are derived from potential energy functions. Given a condition $C(\mathbf{p})$, dependent on a set of nodes $\{n_1, \ldots, n_c\}$ with positions $\mathbf{p}_1, \ldots, \mathbf{p}_c$, $\mathbf{p} = \{\mathbf{p}_1, \ldots, \mathbf{p}_c\}$, we require this condition to be zero. The associated energy function is:

$$E_C = \frac{1}{2} k C(\mathbf{p})^2,$$

where $k$ is a stiffness constant. The force is given as a negative gradient of the energy function, with respect to the positions of the nodes:

$$\mathbf{F}_{E,i} = -\frac{\partial E_C}{\partial \mathbf{p}_i} = -k C(\mathbf{p}) \frac{\partial C(\mathbf{p})}{\partial \mathbf{p}_i}.$$  \hspace{1cm} (4.2)

Applying these forces therefore reduces the deformation energy of the object. Defining damping of the model by energy functions is undesirable as it leads to nonrealistic results. The damping force is therefore defined as in [Baraff and Witkin, 1998]:

$$\mathbf{F}_{D,i} = -k_D \dot{C}(\mathbf{p}) \frac{\partial C(\mathbf{p})}{\partial \mathbf{p}_i}; \quad \dot{C}(\mathbf{p}) = \left( \frac{\partial C(\mathbf{p})}{\partial \mathbf{p}_i} \right)^T \dot{\mathbf{p}}.$$  \hspace{1cm} (4.3)

This damping acts in the same direction as the elastic force is applied, i.e. $\frac{\partial C(\mathbf{p})}{\partial \mathbf{p}_i}$. Moreover, it is proportional to the system’s velocity projected to the same direction.
To define the constraint type energy functions for the elastic forces, we use the approach of Teschner et al. [2004]. Two types of forces will be presented: linear spring forces acting between nodes and volume preserving forces defined for each tetrahedra. Adding the volume preserving forces enhances the realism of modelling large deformations.

The condition for the spring force penalises the difference of the current edge length $\|\text{EDGE}_{ij}\| = \|\mathbf{p}_i - \mathbf{p}_j\|$, between two nodes $n_i$ and $n_j$, to its initial length $l_0$:

$$C_S(\text{EDGE}_{ij}) = \|\mathbf{p}_i - \mathbf{p}_j\| - l_0.$$  \hspace{1cm} (4.4)

The elastic spring force $\mathbf{F}_{S,ij}$, and the damping force $\mathbf{F}_{D,ij}$, defined for the spring $S = \{i,j\}$ and acting at the vertex $i$ are:

$$\mathbf{F}_S^i = -k_S (\|\mathbf{p}_i - \mathbf{p}_j\| - l_0) \frac{(\mathbf{p}_i - \mathbf{p}_j)}{\|\mathbf{p}_i - \mathbf{p}_j\|},$$  \hspace{1cm} (4.5)

$$\mathbf{F}_D^i = -k_D (\mathbf{v}_i - \mathbf{v}_j) \cdot \mathbf{(p}_i - \mathbf{p}_j) \frac{(\mathbf{p}_i - \mathbf{p}_j)}{\|\mathbf{p}_i - \mathbf{p}_j\|} \frac{(\mathbf{p}_i - \mathbf{p}_j)}{\|\mathbf{p}_i - \mathbf{p}_j\|},$$  \hspace{1cm} (4.6)

where $k_S$, $k_D$ define the spring stiffness and spring damping, respectively, $\mathbf{p}_i$, $\mathbf{p}_j$ denote the positions of the nodes $i$ and $j$, and $\mathbf{v}_i$, $\mathbf{v}_j$ denote the velocities of the nodes $i$ and $j$.

Additionally, the volume preserving forces are defined for a constraint considering the absolute difference of the current volume of the tetrahedra $\text{TET}_{ijkl}$ and its initial volume $V_0$:

$$C_V(\text{TET}_{ijkl}) = \frac{1}{6} (\mathbf{p}_i - \mathbf{p}_j) \cdot ((\mathbf{p}_l - \mathbf{p}_j) \times (\mathbf{p}_k - \mathbf{p}_j)) - V_0,$$  \hspace{1cm} (4.7)

where the current volume of the tetrahedra is expressed by the mixed product rule. The force $\mathbf{F}_V^i$, applied at node $i$ of the tetrahedron $\text{TET}_{ijkl}$, which penalises the change in volume of the tetrahedron $\text{TET}_{ijkl}$, is therefore:

$$\mathbf{F}_V^i = -\frac{k_V}{6} \left( \frac{1}{6} (\mathbf{p}_i - \mathbf{p}_j) \cdot ((\mathbf{p}_l - \mathbf{p}_j) \times (\mathbf{p}_k - \mathbf{p}_j)) - V_0 \right) \left( ((\mathbf{p}_l - \mathbf{p}_j) \times (\mathbf{p}_k - \mathbf{p}_j)) \right),$$  \hspace{1cm} (4.8)

where the last term the vector $((\mathbf{p}_l - \mathbf{p}_j) \times (\mathbf{p}_k - \mathbf{p}_j))$ is perpendicular to the face $\text{TRI}_{jkl}$, and its magnitude represents twice the area of the face opposite to node $i$.

The Equation 4.8 can be symbolically rewritten as:

$$\mathbf{F}_V^i = -\frac{k_V}{6} \left( \text{Volume}(\text{TET}_{ijkl}) - V_0 \right) \left( 2 \cdot \text{Area}(\text{TRI}_{jkl}) \right).$$

Here, the $\text{Volume}(.)$ function returns a signed volume of the tetrahedron, as it is calculated with the mixed product in Equation 4.7, and the $\text{Area}(.)$ is a function returning a vector, given by the outer product used in Equation 4.8.
Note, that since we use absolute (and not relative) difference in distance or volume calculation, the stiffness coefficients for the defined forces are not scale invariant.

One of the problems when using MSSis that the springs do not maintain their orientation with respect to the neighbouring nodes. It happens that under high load the tetrahedra invert, in order to relieve the strain applied at the springs. This demonstrates itself as if the mesh would collapse. The application of volume preserving forces mitigates this effect, as they depend on the positions of more than two points. Especially, the signed volume of the constraint function, as it is calculated in Equation 4.7, is of major importance. It detects inversions of the tetrahedra’s orientation, in case some of its edges flipped, and gives rise to forces which restore the initial orientation of the tetrahedron (see Figure 4.4.1 for an example of applied forces for one tetrahedron).

\[ F_{\text{internal}} = - (F_S + F_D + F_V). \] (4.9)

Here \( M \) is the mass matrix of the model, \( D \) is the nodal damping matrix, and \( F \) represents internal or external forces, respectively. The internal forces can be expressed in terms of the spring forces and volume preserving forces (see also Equations 4.5, 4.8):

\( F_{\text{internal}} = - (F_S + F_D + F_V). \)

The damping matrix \( (D) \) represents viscous damping applied at each node of the mesh. For MSSs the matrices \( M \) and \( D \) are diagonal.
with entries $m_i$ and $d_i$, respectively. To update the node positions and velocities, we applied explicit numerical integration schemes. These are explained in detail in Section 4.4.3.

### 4.4.2 Parameters setting of the Mass-Spring system

Stable and realistic behaviour of the MSS depends strongly on the parameters of the model. These include the mesh geometry, stiffness of the inner forces (i.e. forces applied by the springs and by the volume constraints), damping parameters and masses of the nodes.

#### Mesh generation

Stable and real-time simulation of mesh-based deformation models depends on the quality of the underlying mesh. Big variance in the mesh structure (i.e. edge lengths, angles between the edges or volumes of the mesh elements) results in a low quality mesh, for which stiff deformation equations will be obtained. These are hard to be solved numerically.

To define the spring connections, volume preserving forces and mass distribution, we use a tetrahedral mesh of the deformable body. Our meshes were created with a simple mesh generator presented by Persson and Strang [2004]. Throughout the mesh construction, piecewise linear force-displacement relations between the mesh nodes are applied in order to solve for the positions of the vertices. Additionally, Delaunay triangulation is employed to set the topology of the edges. This generator produces meshes of high quality, suitable for computer graphics as well as for scientific computing.

#### Parameters of the internal forces

In the following we will present several methods which have been used for the parameter setting of the MSS, and which we adopted for our simulations. Since these methods are based on a static analysis of object’s deformations, it is not possible to obtain the parameters for the damping forces (the viscous nodal damping as well as the spring damping). Therefore, the node and spring damping parameters have been set to a constant value which has been manually tuned to ensure stable dynamic deformation.

MSSs have been used in many applications, however, only little attention was given to the setting of parameters resulting in realistic behaviour. Typically, the parameters are set manually by tuning the stiffness, damping and viscosity parameters for
all springs and nodes. Often, all springs are set with equal parameter values to ease this cumbersome process. Good quality settings are expected to provide pleasing visual appearance of the deformations.

Two other approaches have been described recently. The first one is based on an optimisation process, which tries to adapt the deformation behaviour of the MSS to a reference model. The reference data usually include several deformation states, which were either measured on real samples or generated offline, using a model based on constitutive equations with known material parameters. For example, Deussen et al. [1995] have used simulated annealing optimisation to obtain good approximations of the spring parameters. For the optimisation problem, the authors have considered nine basic loads to obtain the reference configurations. The quality of the mesh has been measured by the standard deviation between the actual and the reference displacements of all points. Different optimisation techniques, like adapted simulated annealing [Morris, 2006], neural networks [Nürnberger et al., 1998] and genetic algorithms [Joukhadar et al., 1997; Bianchi et al., 2004] were also explored.

The data-driven approaches often lead to exponential growth of the optimisation complexity for 3D objects with complex topology. Moreover, the obtained parameters are object specific. Therefore, the calibrated constants cannot be transferred between objects of the same material.

The second strategy, an analytical approach in parameter identification, particularly addresses the drawbacks of the previous methods. It relates the spring parameters to the constitutive material parameters. Therefore, simple calibration of the deformation parameters is possible for objects of different geometry and material. The idea behind this method is to describe the deformations of the flexible body using a model based on constitutive equations (e.g. FEM) and extract the parameters by a comparison of corresponding structures. Specifically, the stiffness matrices of the constitutive model and the one of the MSS are examined. However, a theoretical proof was presented by [Gelder, 1998], showing that simply equating the two matrices is, in general, not possible due to differences in zero elements. This demonstrates, that a MSS cannot exactly represent the deformations of a constitutive FEM. Nevertheless, a geometrically based relation was derived by Gelder [1998], which defines plausible spring stiffness parameters for triangular MSSs. This result was heuristically extended to tetrahedral meshes.

Based on the Young’s modulus $E$ of the material, Gelder [1998] expresses the stiffness of each spring, defined by an edge $e$, by:

$$ k_e = \frac{E \sum_{\text{TET} \ni e} \text{Volume(TET)}}{\text{Length}(e)^2}. $$

(4.10)

Here the summation goes over all tetrahedra which include the edge $e$. Setting the spring parameters according to Equation 4.10 yields satisfactory response to applied
loads, in the sense of modelled deformations. This has been shown for applications in computer graphics [Gelder, 1998] as well as in surgical simulation [Paloc et al., 2002].

To complete the parameter setting for our MSS we need to define the stiffness of the volume preserving forces, defined in Section 4.4.1. Unfortunately, it is not clear, how the idea presented in [Gelder, 1998] could be applied for determining all the required stiffness parameters.

Another technique for determining the deformation parameters by exploiting the structure of the stiffness matrices was given by Cavusoglu [2000]. To avoid the correspondences of zero elements, Cavusoglu [2000] proposed to find the stiffness parameters by minimising the difference of the stiffness matrices of the Finite element method ($K_{FEM}$) and the corresponding Mass-Spring system ($K_{MSS}$) in some norm:

$$k_S = \arg \inf_{k_S} \|K_{FEM}(E, \nu) - K_{MSS}(k_S)\|.$$  \hspace{1cm} (4.11)

Here, the stiffness matrix of the MSS could easily include the volume preserving forces. The expressions for the stiffness matrix $K_{MSS}$ can be found in Appendix A. For the derivation of the matrix $K_{FEM}$ one can see [Alberty et al., 2002].

The minimisation step was proposed to be performed for every different element configuration of the mesh, as the result depends on the geometry of the given element. However, during the optimisation process applied on arbitrary meshes, negative stiffness parameters can be obtained\(^2\) for certain geometries of the mesh elements. A setting with negative stiffness parameters is physically not plausible and leads to instabilities of the modelled deformations.

Recently, Lloyd et al. [2007] followed on the idea of using optimisation techniques to minimise the difference between the stiffness matrices in order to provide plausible parameter settings for tetrahedral MSS. Volume preserving forces, as have been defined in Section 4.4.1, were also considered in the study.

The authors observed, that for regular tetrahedra the obtained relation yields non-negative results. Therefore, to avoid the occurrence of negative stiffness parameters, they consider every tetrahedra to be regular and of the same volume as the original

\(^2\)The negative stiffness parameters are obtained due to the different structure of the stiffness matrices defined by the FEM system and the springs of the MSS. Adding volume preserving forces does not influence the parameter setting process.
one. Using this assumption, the relations for spring as well as volume preserving stiffness parameters were given:

\[
k_S = \frac{2 \sqrt{2}}{21} \sum_{TET \ni e} E \hat{\ell}_e \frac{4}{5},
\]

(4.12)

\[
k_V = \frac{2}{35} V_{TET}^3 E,
\]

(4.13)

for the stiffness parameters of the springs \( k_S \) and the volume preserving forces \( k_V \). Here, \( \hat{\ell}_e \) denotes the length of the edge of a regular tetrahedra. The equivalent edge length for a nonregular tetrahedra of volume \( V_{TET} \) is computed by:

\[
\hat{\ell}_e = \left( \frac{12}{\sqrt{2}} V_{TET} \right)^{1/3}.
\]

(4.14)

Comparing several object deformations to a reference, linear FEM, Lloyd et al. [2007] showed that better results are obtained if volume preserving forces are applied in addition to the spring forces. The expression introduced in Equation 4.13 gives the optimal stiffness values for the volume preserving forces. Unfortunately, setting the suggested parameters is not strong enough to avoid inverting of the tetrahedra when high load is applied. From practical experiments we found that the volume preserving forces must be stiffer by a factor of seven and more in order to keep the mesh geometry realistic. The reason for this is most probably in different boundary conditions for which the relations were derived and evaluated. The tetrahedra gets inverted only under high loads, which lead to big deformations. These cannot be modelled realistically with the linear FEM, which was used for analytically obtaining the deformation parameters. Also, different stability criteria of our dynamically simulated MSS, when compared to the static analysis used in [Lloyd et al., 2007], can influence the applicability of the obtained results.

At this moment we can only conclude that a fully automatic parameter setting for our MSS remains still an open issue. To set up the physical properties of our system we therefore adjust manually the compliance of the virtual object. However, we use only two elasticity parameters. One represents the elasticity modulus of the springs and the other one the elasticity of the volume preserving forces. We first set the distribution of the spring stiffness parameters according to Equation 4.10 or Equation 4.12. Similar, for the volume preserving forces we either consider an uniform distribution or the one obtained by Equation 4.13. Using the relations from the aforementioned studies we assure that the stiffness parameters will account for the different geometry of the tetrahedra. Afterwards we only have to tune the two elasticity moduli according to which the specific parameters of each spring and tetrahedra are set. In general, we increase the stiffness of the volume preserving forces until no mesh collapse can be observed during a typical interaction. Since
this choice makes the virtual object harder, we consecutively reduce the elasticity modulus of the springs. The precision of the resulting stiffness can be verified by comparing several deformation states to given ground truth configurations.

Masses

Setting up the masses of the nodes is done by redistributing the overall mass of the simulated object according to Deussen et al. [1995]. Mass moments, of the discretised body up to the second order are matched to exact mass moments of the simulated object, corresponding to given geometry and mass distribution. The condition on the mass moments, for a mass density function $\rho(\vec{u})$, is:

$$\sum_{i=1}^{n} p_{i,x}^j p_{i,y}^k p_{i,z}^l m_i = \int_{\text{Body}} u_x^j u_y^k u_z^l \rho(\vec{u}) \, d\vec{u} , \quad 0 \leq j + k + l \leq 2 ,$$

where the mass of each node is $m_i$ and its position is $p_i$. Hence, in matrix form we can write:

$$A \begin{bmatrix} m_1, \ldots, m_n \end{bmatrix}^T = \begin{bmatrix} b_{000}, b_{100}, b_{010}, b_{001}, b_{110}, b_{011}, b_{200}, b_{020}, b_{002} \end{bmatrix}^T ,$$

$$b_{jkl} = \int_{\text{Body}} u_x^j u_y^k u_z^l \rho(\vec{u}) \, d\vec{u} .$$

The last equation defines an under-determined system (supposed $n > 10$) of ten equations. To solve this system, we first estimate the mass of each node $n_i$, using the volume of the incident tetrahedra as a weighting factor for distributing the overall mass of the object. Then we apply least-squares minimisation of the unknown masses $m_i$ to the estimated values, while enforcing the condition on the mass moments:

$$\min \left( \sum_i \left( m_i - \rho \frac{1}{4} \sum_{\text{TET} \ni n_i} \text{Volume(TET)} \right)^2 \right) , \quad \text{such that} \quad Am = b$$

where $\rho$ is the density of the material and TET is a tetrahedron defined by the model mesh.

The exact mass moments, necessary for this optimisation, were computed analytically for simple shapes. For objects, whose geometry was too complex to get analytical expressions of the mass moments, we only used the values estimated by the volume of incident tetrahedra:

$$m_i = \rho \frac{1}{4} \sum_{\text{TET} \ni n_i} \text{Volume(TET)} .$$

This solution is a good approximation of the true mass distribution.
Note that the mass distribution is important for a dynamic simulation of the object’s deformations. Nevertheless, other factors exist, which influence the realism of a dynamic simulation. These are the damping parameters, set for each node and for each spring of the model. A proper distribution of the masses can not be achieved without an appropriate method of measuring and setting the damping parameters.

### 4.4.3 Numerical integration

Given the positions $p_i(t)$ and velocities $v_i(t)$ at time $t$, we want to determine the new positions $p_i(t + \Delta t)$ and new velocities $v_i(t + \Delta t)$ at time $t + \Delta t$, which would satisfy the equation of motion of the MSS (Equation 4.9). We first transform the Equation 4.9 into a first order differential equation:

$$\frac{\partial}{\partial t} \begin{pmatrix} p \\ Mv \end{pmatrix} = \begin{pmatrix} v \\ F_{\text{external}} - Dv - F_{\text{internal}}(p, v) \end{pmatrix}.$$  (4.18)

In the following we use $F_E = F_{\text{external}}$ to express the external forces applied, while the internal forces are given by the term (see Equations 4.5, 4.6 and 4.8):

$$F(p, v) = F_S(p, v) + F_D(p, v) + F_V(p, v) = -F_{\text{internal}}(p, v).$$

The internal forces are functions dependent on the node positions and velocities, $p$ and $v$. To simplify the notation, we omit the variables of the internal force function $F = F_S + F_D + F_V$ and denote the internal force acting at the node $i$ by $F_i$. Also, we denote the time parameter in the superscript of each variable (i.e. $p_{t}^{i}$, $v_{t}^{i}$ and $F_{t}^{i}$).

Now we present several algorithms, which solve the Equation 4.18 numerically. The solvers will be initialised by the starting positions, $p_{0}^{i}$, and velocities, $v_{0}^{i}$. Given the applied external forces at each time step $t$, a piecewise linear trajectory for each mass point is computed.

**Explicit integration**

The simplest method of numerical integration is the **explicit Euler** scheme, which uses the positions, velocities and external forces at the current time step to compute the corresponding values for the following time step. The explicit Euler scheme is given by the rules:

$$\begin{align*}
    a_{t}^{i} &= \frac{F_{E,i}^{t} - d_{i}v_{t}^{i} + F_{i}^{t}}{m_{i}} \\
    p_{t}^{i+\Delta t} &= p_{t}^{i} + \Delta t v_{t}^{i} \\
    v_{t}^{i+\Delta t} &= v_{t}^{i} + \Delta t a_{t}^{i},
\end{align*}$$  (4.19)
where $a_i$ expresses the accelerations of the node $i$ due to the applied internal and external forces. This scheme guarantees only first order precision, $O(\Delta t)$, of the computed positions and velocities. More precise results can be obtained using the Velocity Verlet scheme, which ensures second order precision, $O(\Delta t^2)$, for the positions as well as velocities:

$$a_i^t = \frac{F_{E,i}^t - d_i v_i^t + F_i^t}{m_i}$$

$$p_i^{t+\Delta t} = p_i^t + \Delta t v_i^t + \Delta t^2 \frac{a_i^t}{2}$$

$$v_i^{t+\Delta t} = v_i^t + \Delta t \left( a_i^t + a_i^{t+\Delta t} \right).$$

This scheme has been very popular, because it provides plausible results and a good ratio between the maximum possible time step and the computation time if compared to other explicit integration methods [Teschner et al., 2004].

For our application, however, the computation of forces $F_i^{t+\Delta t}$ in the Equation 4.20 poses a problem. Due to the use of spring damping, the velocities $v_i^{t+\Delta t}$ are involved in the computation of the inner forces at the time step $t + \Delta t$. We therefore applied a slight variation of the Velocity Verlet method:

$$a_i^t = \frac{F_{E,i}^t - d_i v_i^t + F_i^t}{m_i}$$

$$v_i^{t+\Delta t} = v_i^t + \Delta t \left( a_i^t + a_i^{t-\Delta t} \right)$$

$$p_i^{t+\Delta t} = p_i^t + \Delta t v_i^{t+\Delta t} + \Delta t^2 \frac{a_i^t}{2}.$$

The modification of this algorithm lies in that we compute and use the positions one time step ahead in contrast to the velocities. Nevertheless, this influences only the computation of the damping forces. At each time step, velocities $v_i^{t-\Delta t}$ are involved in the computation of the internal and viscosity forces, present in $F_i^t$.

Within our simulation framework we also compared the above schemes to other explicit integration approaches, like the Beeman method or the Newmark scheme. We experienced the best results with the modified version of the Velocity Verlet technique. We have to mention, that our criteria were the stability, fast update and realistic visual appearance of the deformations.

**Implicit integration**

The general problem of using explicit integration schemes is that a stable behaviour can be only reached for very small time steps. In contrast to this, implicit integration schemes enable unconditionally stable integration, allowing for the use of
large time steps during the simulation. The implicit schemes are solving for the next state, whose derivative points to the current state of the object. Therefore, the state variables from the next time step, i.e. \( p^{t+\Delta t} \) and \( v^{t+\Delta t} \), are employed in the computation. One of the most simple implicit integration schemes is the backward Euler method, given by:

\[
\begin{align*}
    v^{t+\Delta t} &= v^t + \Delta t \frac{F_E - d v^{t+\Delta t} + \mathbf{F}^{t+\Delta t}}{m} \quad (4.22) \\
    p^{t+\Delta t} &= p^t + \Delta t v^{t+\Delta t}. \quad (4.23)
\end{align*}
\]

To compute the unknown force \( \mathbf{F}^{t+\Delta t} \), a first order approximation by the Taylor series is used:

\[
\mathbf{F}^{t+\Delta t} = \mathbf{F}^t + \frac{\partial \mathbf{F}^t}{\partial \mathbf{p}} (p^{t+\Delta t} - p^t) + \frac{\partial \mathbf{F}^t}{\partial \mathbf{v}} (v^{t+\Delta t} - v^t). \quad (4.24)
\]

We can now use Equation 4.23 to express the implicit integration step for the velocities:

\[
\begin{align*}
    \hat{\mathbf{M}} v^{t+\Delta t} &= \Delta t \hat{\mathbf{F}} , \quad (4.25) \\
    \hat{\mathbf{M}} &= \mathbf{M} + \Delta t \mathbf{D} - \Delta t \frac{\partial \mathbf{F}^t}{\partial \mathbf{v}} - \Delta t^2 \frac{\partial \mathbf{F}^t}{\partial \mathbf{p}}, \\
    \hat{\mathbf{F}} &= \mathbf{F}_E + \mathbf{F}^t + \left( \frac{\mathbf{M}}{\Delta t} - \frac{\partial \mathbf{F}^t}{\partial \mathbf{v}} \right) v^t.
\end{align*}
\]

Note that in the above equation we assume the external forces \( \mathbf{F}_E \) to be constant during each time step, therefore the partial derivatives of the external forces with respect to positions or velocities vanish.

The terms \( \frac{\partial \mathbf{F}^t}{\partial \mathbf{p}} \) and \( \frac{\partial \mathbf{F}^t}{\partial \mathbf{v}} \) are the Jacobians of the internal elastic and damping forces, respectively, and will be denoted \( J_{p,ij} = \frac{\partial \mathbf{F}^t}{\partial p_i} \) and \( J_{v,ij} = \frac{\partial \mathbf{F}^t}{\partial v_j} \). The precise formulation of the Jacobians is given in Appendix A and Appendix B.

For the elastic forces, the Jacobians are often referred to as the negated Hessians of the MSS [Desbrun et al., 1999]. This is because the elastic forces are defined as a negated gradient of a corresponding energy function. For the damping of the system, it would not be correct to speak about Hessians of the system, as the damping forces were defined heuristically.

In Equation 4.25, the integration scheme is expressed in matrix notation. Because the equations of motions for the mass nodes are coupled, the involved matrices are not diagonal. To determine the unknown velocities, a linear system of equations has to be solved at each iteration step. Using implicit integration, therefore, involves a linearisation of the applied forces, i.e. construction of the Jacobian matrices, at each time step. Since this poses a big computational burden for the application, we decided to use an approximate method to assure fast update rates.
Semi-implicit integration

In the following we will show how to decouple the system of equations 4.25, to circumvent the time consuming building of the Jacobian matrices and solving of the linear system. During this process, several simplifications will be made, which will degrade the implicitness of the scheme described by Equation 4.25. Therefore, we will refer to the following scheme as semi-implicit integration method.

First, we exploit the sparsity of the Jacobians $J_p$ and $J_v$. They are nonzero only for indices $i,j$ where the nodes $n_i, n_j$ belong to the same edge (i.e. they also belong to the same tetrahedra) of the mesh. Furthermore we recall, that the matrices $M$ and $D$ are always diagonal for the MSSs. The update rule for each node can therefore be rewritten as:

$$\hat{M}_{ii} v_i^{t+\Delta t} = \Delta t \hat{F}_i,$$

$$\hat{M}_{ii} = m_i I + \Delta t d_i I - \Delta t J_{v,ii} - \Delta t^2 J_{p,ii},$$

$$\hat{F}_{p,i} = F_{E,i} + F_{t,i} + \left(\frac{m_i}{\Delta t} I - J_{v,ii}\right) v_i^t - \sum_{j \in \mathcal{N}(i)} J_{v,ij} v_j^t +$$

$$+ \sum_{j \in \mathcal{N}(i)} (J_{v,ij} + \Delta t J_{p,ij}) v_j^{t+\Delta t}$$

where $\mathcal{N}(i)$ denotes the neighbours of the node $n_i$.

Finally, to express the above equation explicitly, we will use an approximation of the velocities $v_j^{t+\Delta t}$ at the next time step as proposed by Kang et al. [2000]. The prediction of the velocity change $\Delta v_j^{t+\Delta t} = v_j^{t+\Delta t} - v_j^t$ for the next time step is based on $\Delta v_j^t$ and the currently applied forces:

$$\Delta v_j^{t+\Delta t} \approx \|\Delta v_j^t\| \frac{F_{E,j} + F_{t,j}'}{\|F_{E,j} + F_{t,j}'\|}.$$

The authors explain this relation by an assumption, that while the magnitude of the velocity change does not differ much between the time steps, the direction of the velocity change coincides with the direction of the forces applied at the particle.

Using Equation 4.27, it would be possible to express the update step for each particle (Equation 4.26) explicitly. However, building all involved Jacobians of the internal forces, i.e. the spring, damping and volume preserving forces as defined in Section 4.4.1, would still pose high computational demand. The update rate would be similar as for the implicit integration scheme, although solving a linear system could be avoided. Nevertheless, we found, that it is sufficient to approximate the Jacobians of the internal force functions by using a simplified definition of the internal forces.
Following Desbrun et al. [1999], we replaced the elastic spring force, defined for the spring $S = \{i, j\}$ and acting at node $i$,

$$F^i_S = -k_S (p_i - p_j) + k_S l_0 \frac{(p_i - p_j)}{\|p_i - p_j\|},$$

by its linear part:

$$F^i_S = -k_S (p_i - p_j).$$

Although this defines forces which would act in the MSS if all rest lengths of the springs were zero, we use this approach only for expressing the Jacobians of the spring forces which are involved in the numerical integration. Neglecting the second, nonlinear part introduces errors in the rotation of the particles.

Further, we used simple spring damping, for which the relative velocity of the spring nodes is not projected at the spring direction. Given a spring $S = \{i, j\}$, the damping force applied at node $i$ is:

$$F^i_D = -k_D (v_i - v_j).$$

At last, we neglected the contribution of the volume preserving forces to the Jacobians of the internal forces. Thus, the volume preservation forces will be involved only in the prediction step (Equation 4.27) and in the computation of internal forces $F^t_i$. Now, the Jacobians $J_p$ and $J_v$ can be easily expressed by:

$$J_{p,ii} = - \sum_{\forall S = \{i,j\}} k_S I,$$

$$J_{p,ij} = k_S I, \quad \text{for } S = \{i,j\},$$

$$J_{v,ii} = - \sum_{\forall S = \{i,j\}} k_D I,$$

$$J_{v,ij} = k_D I, \quad \text{for } S = \{i,j\}.$$  \hspace{1cm} (4.28)

Using the presented approach we were able to obtain stable simulation of the deformations for large time steps, without increasing the computational burden.

### Setting the time step

During our experiments we made the best experience using the explicit Velocity Verlet (Equation 4.21) and the Semi-Implicit (Equation 4.26) schemes. Nevertheless, the setting of the time step differs for both methods.

We find the simulation time step for the explicit scheme according to Delingette [1998], using:

$$\Delta t \approx \sqrt{\frac{m}{n \pi^2 k_{\text{max}}}},$$

where $k_{\text{max}}$ is the maximum spring stiffness, $n$ is the number of nodes, and $m$ is the total mass of the deformable object. This equation estimates the time step
4.4. Components

beyond which the system of equations of motion is divergent when using an explicit numerical integration method. We observed good results using this approximate value. Nevertheless, since in our model we also apply damping forces, this time step can be increased.

To choose the right time step for the semi-implicit scheme we follow [Desbrun et al., 1999]. For the simplified formulation (see Equation 4.28), the Jacobians $J_p$ and $J_v$ in Equation 4.25 have a zero eigenvalue for the eigenvector $(1, 1, \ldots, 1)^T$. As a consequence, the matrix $\frac{1}{m_i + \Delta t d_i} \hat{M}$ has an eigenvalue of 1 for the same eigenvector. Therefore, computing the new velocity values by solving the linear system of equations equals to filtering of the applied forces by the matrix:

$$
\left( \frac{1}{m_i + \Delta t d_i} \hat{M} \right)^{-1} = \left( I - \frac{\Delta t}{m_i + \Delta t d_i} J_v - \frac{\Delta t^2}{m_i + \Delta t d_i} J_p \right)^{-1}.
$$

Here, the Jacobians depend only on the stiffness and the damping parameters. Setting the time step very small, i.e. $(\Delta t k_D + \Delta t^2 k_S) \ll 1$, the integration will be equivalent to an explicit one, with external forces affecting only the intermediate neighbours within a single step. For large time steps, however, almost a constant filtering will be applied. This results in a rigid body like motion, where all nodes are translated together.

We noticed, that for our application the semi-implicit scheme is stable, if no volume preserving forces are applied. Adding the volume preserving forces introduces strong vibrations of all nodes for large time steps. This is because we did not include the Jacobians of the volume preserving forces into the velocity update step. Although this behaviour presents a drawback of the simplified method, it helps us in choosing the right time step. We find the suitable time step empirically, by choosing the value for which no vibrations are noticeable.

Comparison of the numerical integration schemes

In order to optimise the performance of our system, we have compared the speed and real-time performance of the integration methods presented in this section on several simple scenarios. As a performance measure we have chosen the ratio between the maximum stable time step and average time needed for one update step. Nevertheless, for a stable haptic interaction not only the real-time ratio, but also a fast update rate is one of the crucial factors. The fast refresh rate together with small time step of the integration scheme minimises the discontinuities of the node positions, and thus generates only little changes of the body surface. This enhances the stability of the haptic interaction with the soft body.

Three virtual objects, a block and two cylinders of a different resolution, selected for the test, are shown on Figure 4.3, and their parameters are summarised in
Figure 4.3: The virtual objects used for the comparison test. From left to right: block, cylinder1 and cylinder2.

Table 4.1. In Table 4.2 the refresh rate, time step and real-time ratio observed for different methods while interacting with the test objects are shown.

<table>
<thead>
<tr>
<th>setup</th>
<th>dimensions</th>
<th>nodes</th>
<th>springs</th>
<th>tetrahedra</th>
<th>density</th>
<th>elasticity</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>$10 \times 5 \times 10 \text{ mm}$</td>
<td>196</td>
<td>1017</td>
<td>678</td>
<td>995 kg/m$^3$</td>
<td>15 kPa</td>
</tr>
<tr>
<td>cylinder1</td>
<td>$\varnothing 80 \times 80 \text{ mm}$</td>
<td>89</td>
<td>451</td>
<td>295</td>
<td>995 kg/m$^3$</td>
<td>15 kPa</td>
</tr>
<tr>
<td>cylinder2</td>
<td>$\varnothing 80 \times 80 \text{ mm}$</td>
<td>300</td>
<td>1656</td>
<td>1156</td>
<td>995 kg/m$^3$</td>
<td>15 kPa</td>
</tr>
</tbody>
</table>

Table 4.1: The main parameters of the virtual bodies as shown on Figure 4.3. The density and the elasticity represent material parameters of the reference objects, according to which the deformation properties of the virtual bodies were set.

The implicit scheme was not included in the tests, since already during the implementation we found that only the construction of the stiffness matrix would take three times longer than one update step of our simplified semi-implicit method. The low update rates would radically degrade the stability of the haptic interaction.

Among the investigated numerical integration schemes we obtained best results with the Velocity Verlet and the semi-implicit scheme. Note, that although the semi-implicit method allows for much higher time steps than the Velocity Verlet scheme, the simulation shows high level of viscosity added to the movement of the soft body. This is due to the simplifications made. In fact, the additional virtual viscosity is needed to make the semi-implicit scheme (which belongs to a group of explicit schemes) performing stable even for big time steps. In our haptic simulation we therefore use both methods, while the preference is to use the semi-implicit method, unless the virtual viscosity strongly degrades the appearance of the simulation results.
### Table 4.2: Comparison of the performance for different numerical integration schemes.
The time step shows the maximum value for which the interaction was still stable. For the semi-implicit scheme, however, a time step which provides real-time update of the deformable body is given, as the scheme is stable even for very large time steps.

<table>
<thead>
<tr>
<th>Method</th>
<th>Setup</th>
<th>Time Step</th>
<th>Refresh Rate</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit Euler</td>
<td>block</td>
<td>0.0005</td>
<td>124</td>
<td>0.062</td>
</tr>
<tr>
<td></td>
<td>cylinder1</td>
<td>0.0011</td>
<td>285</td>
<td>0.314</td>
</tr>
<tr>
<td></td>
<td>cylinder2</td>
<td>0.00015</td>
<td>77</td>
<td>0.011</td>
</tr>
<tr>
<td>Velocity Verlet</td>
<td>block</td>
<td>0.00085</td>
<td>113</td>
<td>0.095</td>
</tr>
<tr>
<td></td>
<td>cylinder1</td>
<td>0.0017</td>
<td>261</td>
<td>0.444</td>
</tr>
<tr>
<td></td>
<td>cylinder2</td>
<td>0.00035</td>
<td>73</td>
<td>0.025</td>
</tr>
<tr>
<td>Beeman</td>
<td>block</td>
<td>0.00045</td>
<td>80</td>
<td>0.035</td>
</tr>
<tr>
<td></td>
<td>cylinder1</td>
<td>0.0008</td>
<td>183</td>
<td>0.148</td>
</tr>
<tr>
<td></td>
<td>cylinder2</td>
<td>0.00015</td>
<td>47</td>
<td>0.007</td>
</tr>
<tr>
<td>Semi-Implicit</td>
<td>block</td>
<td>&gt; 0.0101</td>
<td>102</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>cylinder1</td>
<td>&gt; 0.0043</td>
<td>231</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>cylinder2</td>
<td>&gt; 0.0172</td>
<td>61</td>
<td>1.0</td>
</tr>
</tbody>
</table>

#### 4.4.4 Collision detection

Finding collisions between deformable objects is difficult, since the geometrical shape of the body changes if subjected to an external force. Due to this fact, any precomputed data, which could speed up the detection cannot be used efficiently as they have to be updated after each simulation step. Therefore, we have chosen a simple approach for collision detection, based on distance information computed for a fixed set of nearest features. As a feature we consider a pair of points, each of which belongs to two distinct objects.

The distance information, carried by each of the features, will be updated by tracking the nearest points for each feature. We will, therefore, refer to this algorithm as to tracking of nearest features. The update of the features will be processed locally, which warrants low computational complexity of the method. As a consequence, the update of the nearest features can be run within the fast haptic thread. The most recent distance information can then be used for computing the contact forces, which will enhance the stability of the interaction.

For the purpose of our application, we simplify the strategy for tracking the nearest features by fixing one point of each feature, namely the point belonging to the virtual tool (see Figure 4.4). For the update step we will therefore track the nearest points on the surface of the deformable body. The collision detection algorithm will thus
detect collisions between the surface of the deformable object and a certain fixed set of points at the rigid tool. More precisely, we will detect collisions between vertices defined at the tool and surface triangles of the mesh representing the soft body. For smooth detection of contacts it is therefore important to densely cover the tool with the set of feature points.

Our approach for tracking nearest features is similar to the one presented by Ho et al. [1999], who is monitoring the transitions of the nearest surface point between the neighbouring surface primitives. These include face to edge or vertex, edge to vertex or face and vertex to edge or face transitions. Nevertheless, it differs in the use of lower complexity data structure and in reduced number of distance queries, when searching the neighbouring primitives (i.e. vertex, edge or triangle) for the nearest one to the haptic interaction point.

Moreover, the application of this method also differs. While Ho et al. [1999] have used their approach solely for the collision response algorithm, i.e. tracking the nearest point on the surface once the virtual tool collided with the object, we apply the same technique for collision detection as well. This decision poses restrictions on the shape of the virtual bodies - i.e. convexity. However, within our scenarios, and for the purpose of investigating haptic rendering algorithms, this choice presents a reasonable trade-off between robust collision detection, low computational requirements and little implementation efforts.

The idea of tracking nearest features for collision detection between arbitrarily shaped rigid objects, represented by a polygonal model, was also proposed by Johnson and Willemsen [2004]. As an intermediate step between a global update of the collision information, the local minimum distances (i.e. nearest features according to our notation) are adjusted over the triangles in the neighbourhood of both feature points. However, more sophisticated strategies are used for managing the set of features.
active features in the global search (i.e. when introducing new features or removing out-dated ones).

**Tracking the nearest features**

At the initialisation step the nearest features are determined by a global search over all surface triangles of the deformable body, for each predefined point of the virtual tool.

Tracking the nearest point on the triangle surface, is done by traversing the neighbouring triangles of the currently nearest triangle, for a given feature, while minimising the distance of the predefined point on the tool to the surface of the soft body. In the following, the currently processed triangle will be denoted as an *actual triangle*. The minimisation process is employing only one query, which is finding the nearest point on a given triangle. The query returns the signed distance to the nearest point on the triangle, indicating contact if the distance is less than zero. If, during the minimisation step, a nearest point is found at the border of the actual triangle, a new actual triangle is set from the neighbouring triangles. This choice is determined during the computation of the distance query for the nearest point. The minimisation step is then repeated until no further local improvements can be achieved.

**Figure 4.5**: Finding the nearest surface point with three special cases: a) the nearest surface point is located at the actual triangle, b) the nearest point of the actual triangle is located at the edge - moving to the neighbouring triangle minimises the distance c) the nearest point of the actual triangle is at one of its vertices - scanning triangles adjacent to the vertex provides a new triangle, for which the distance is minimal.

Three possible cases of the nearest point locations on the actual triangle are considered (see Figure 4.5): in the interior, at the edge or at the vertex of the triangle. The algorithm can be explained in three steps:

a) If the nearest point belongs to the inner area of the actual triangle, we have found the new position of the nearest feature point.
b) If the nearest point belongs to an edge of the actual triangle, we choose the neighbouring triangle sharing this edge to be a new actual triangle for the next step and repeat the process. Crossing the edge to the neighbouring triangle we have to make sure, that we do not cross the edge more than once. In that case the new position of the nearest surface point, is at the edge and we can stop the minimisation procedure.

c) In case the nearest point is located at a vertex of the actual triangle, we start checking the adjacent triangles of the vertex in one direction. Two options are possible; either we find a triangle which reports a new nearest point which is different from the vertex currently under scrutiny, or we return to the actual triangle after checking all adjacent triangles. In the first case we update the actual triangle and repeat the tracking process, in the second case we can stop searching and set the new nearest point, to the examined vertex.

When tracking the nearest point on the triangular mesh, we also have to detect, whether the tool is in contact with the deformable object, for the given feature. This is easily checked by the signed distance measure returned by the query during the search for the nearest point. We only have to handle special cases when the nearest point is found at an edge or at a vertex. For these cases the feature is set to be in contact, if the feature point at the virtual tool is below all adjacent triangles of the edge or the vertex. The contact state of a given feature is therefore updated every time we change to a new actual triangle, according to the three cases which occur during the minimisation search.

The algorithm of tracking nearest features guarantees to check just a minimal number of triangles in order to find the next nearest surface point position. The number of triangles checked depends on the size of the triangles and the distance travelled between the last two collision detection steps. Therefore, it does not depend directly on the size of the object but on the resolution of the deformable object. In addition, since for the haptic rendering algorithm, running at 1 kHz, temporal coherence is very high, the number of triangles crossed is small. This fact, suggesting low computational complexity, is one of the positives of using the tracking algorithm in the fast haptic thread, before the collision force is calculated.

The transition between surface triangles is supported by our data structure (see Figure 4.6). We use a double linked face list, where all triangles keep pointers to their three neighbours in ordered (either clock- or counterclock-wise) manner. The triangles are therefore the main primitives for updating nearest features. To process the distance query when looking for the nearest point, only the vertex positions of each triangle have to be stored in addition.
4.4. Components

Figure 4.6: The data structure showing a list of triangles (T1, T2, T3). The inner data of each triangle includes an ordered list of vertices and correspondingly set pointers to neighboring triangles.

Limitations

Clearly, the presented approach cannot be used for a general collision detection between two objects represented by polyhedral meshes. In our approach we use the simplification of fixing a set of points on the virtual tool. Therefore, only collisions between nodes of the tool mesh with the triangles of the deformable object will be detected. Indeed, a reversed approach could be used to detect the collisions between the triangles of the tool and the vertices of the soft body. In order to detect all collisions between two objects defined by triangle meshes we would have to apply an edge-edge collision detection in addition. This fact poses one of the limitations of our simplified approach. Namely, while we will be able to generate smooth response when touching smooth areas, at sharp edges bigger force discontinuities will appear, leading to a stair-like rendering of the contact forces.

Another limitation applies due to the fact that the algorithm minimises the distance to the tracked points only locally. A correct collision detection cannot be guaranteed for nonconvex objects. For cases when the tool crosses distant surface triangles opposite to the current nearest triangle, the true collision will not be detected (see Figure 4.7). To avoid this, either the tool should follow the surface of the body or a different level of collision detection should be applied in a first stage, which would initiate the tracking of the nearest features on the more distant parts. The latter approach was also applied by Johnson and Willemsen [2004]. Note that, although the collision detection algorithm based on tracking the nearest features may fail for non-convex objects, once in contact, the tracking of the nearest features provides suitable approximation of the penetration distances for each contact point. These distances are used to compute the contact forces within the collision response algorithm.
84

4. System overview

Figure 4.7: Tracking of the nearest feature point: a) normal case (the algorithm can deal with minor non-convexities of the object); b) nonconvex object with a hole - collision was not detected (additional nearest point should be tracked at the other convex part of the object).

The use of the tracking method in the haptic thread limits the number of tracked features. We were able to run the algorithm for around 50 contact points, while keeping the update rate at 1 kHz on a dual Pentium-4 (2.4 GHz) processor PC. However, we did not exploit any parallelism techniques which would speed up the tracking process. Also, clustering the contact points based on their proximity and distribution in the 3D space, as was proposed by Kim et al. [2002], would help to reduce the computational complexity.

4.4.5 Collision response

When the virtual tool is in contact with the deformable object, an external force is applied at the contact point, due to which the soft body is deformed. The computation of the external forces will be presented in Chapter 5. Here, we shortly discuss how the external forces will be applied to the discretised deformation model.

Using MSS the external forces have to be applied just at the mass nodes of the model. Since the real interaction point usually lies in the inner area of a surface triangle, the determined contact force has to be distributed to the nodes of the surface triangle.

We apply two strategies. The first one is to distribute the external force \( \mathbf{F}_E \) based on the barycentric coordinates of the collision point, with respect to the vertex positions of the contacted surface triangle:

\[
\mathbf{F}_{(i,j,k)} = \alpha_{(i,j,k)} \mathbf{F}_E ,
\]  

(4.30)
where $\alpha_{i,j,k}$ are the barycentric coordinates of the collision point with respect to a triangle $\text{TRI}(ijk)$.

Our second approach distributes the external forces to the neighbouring nodes whose distance is within a certain radius of interaction. This approach can be seen as applying Gaussian filtering on non-regular points, for which the weighting of the values is proportional to the distance between the node and the contact point.
In this chapter we describe how the contact force between a rigid tool and a deformable body is computed and consecutively rendered on the haptic device.

Two principles are applied in our systems for the simulation of a virtual tool. The first principle is the paradigm of virtual proxy [Ruspini et al., 1997], which models the interaction of a virtual tool, driven by an input device, with the deformable body. The behaviour of the proxy object in the virtual world will be modelled in static and dynamic manner, according to Newton’s laws of motion. The second principle is the method of virtual coupling [Colgate et al., 1995], which is used to couple the haptic device to the simulation. Using virtual coupling the contact forces returned from the simulation are adapted to the device dynamics and mechanical properties.

In addition, we will use two techniques to compute the contact forces between the proxy object and the soft body: a penalty and a constraint based estimation of the collision forces. We first describe the penalty based model using a single point interaction with an emphasis on achieving smooth, stable and transparent interaction. Filtering and the force shading [Ruspini et al., 1997] techniques will be implemented. Moreover, several friction models will be applied to the technique of the virtual proxy point. Later, the ideas applied to a single interaction point are extended to multiple points in order to simulate 6-DOF interaction of the virtual tool. The straightforward extension of the single point penalty based interaction to 6-DOF interaction with a rigid tool introduces permanent interpenetration of the virtual proxy and the deformable object.

To handle the nonpenetration accurately, a second model, the constraint based technique, will be applied. We compute the collision response forces by solving a constrained dynamic simulation problem, applying Signorini’s contact law and Coulomb’s friction model. This technique reduces the number of parameters necessary for the computation of the contact forces, while providing accurate calculation of the collision forces. Finally, a novel approach, which applies a linearised contact
model with friction within a multirate constraint based simulation will be presented towards the end of this chapter. Using the linearised contact model, highly transparent haptic rendering of contact forces is achieved.

Before giving the details of the force computation models, in the following sections we will shortly revisit the principles of virtual proxy and virtual coupling paradigms. Afterwards we will describe the penalty based method for a single and for multiple contact points (see Sections 5.3 and 5.4, respectively), followed by the constraint based model (see Section 5.5). Section 5.3.3 also includes implementation details of several friction models, which we applied for the penalty based method. Finally, in Section 5.6 we will compare the two proposed contact force computation models, applied for multiple contact points.

5.1 Virtual proxy paradigm

A virtual proxy represents an object in the scene, whose position is driven by the haptic device. More specifically, it is following the movements of the haptic device’s pointer in the virtual world, also noted as the haptic interface tool (HIT). The virtual proxy is aligned with the HIT in free space (see Figure 5.1). When collision occurs, the HIT penetrates the collided body due to technical limitations of the haptic device (i.e. limited refresh rate and maximum force output). The virtual proxy, on the contrary, is constrained to stay on the surface of the collided object.
while locally minimising the distance to the HIT. Thus, it models an ideal position of the HIT interacting with the virtual object.

The distance between the virtual proxy and the HIT, as well as the discrepancy in the rotation between these two objects (in case a 6-DOF interaction is modelled), are used to generate the haptic feedback. In addition, simply by altering the position of the virtual proxy, frictional effects can be modelled and smoothed interaction with polygonal meshes can be obtained.

5.2 Virtual coupling

Virtual coupling (see [Colgate et al., 1995]) is a technique which allows to connect the haptic interface tool to any simulated object in the virtual scene. It generates forces which on one side drive the coupled virtual object and on the other side are sensed by the user operating the haptic device. The force generation routine takes the dynamical and mechanical limits of the haptic device into consideration. Ensuring that the simulation of the coupled body is a discrete time passive system, the virtual coupling adjusts the forces from the virtual world to the haptic device in a way which guarantees the passivity of the haptic environment.

![Figure 5.2](image.png)

**Figure 5.2:** An example of virtual coupling, connecting the haptic interface tool to the virtual object by a linear and angular spring-damper element. Picture taken from [McNeely et al., 1999].

In our application we will apply a viscoelastic virtual coupling as it was used by [McNeely et al., 1999]. It is based on a linear spring-damper element which is linking the HIT with the mass centre of the coupled virtual object (VO). In case a 6-DOF
interaction is modelled, an angular spring-damper element is applied in addition (see Figure 5.2). The virtual coupling attracts the VO and generates coupling forces based on positional ($e_p$) and rotational ($e_\theta$) error measures:

$$e_p = p_{HIT} - p_{VO},$$
$$e_\theta = q_{HIT} q_{VO}^{-1}.$$  \hspace{1cm} (5.1)

Here, $p_{HIT}$ and $q_{HIT}$ represent the haptic interface position and rotation in the virtual world; $p_{VO}$ and $q_{VO}$ represent the position and the rotation of the coupled virtual object. The rotations are expressed in a quaternion notation. Supposed $k_c, k_\theta$ are the stiffness parameters of the linear and the angular springs, respectively, and $b_c, b_\theta$ are the damping parameters of the linear and the angular dampers, the coupling force $F_c$ and torque $T_c$ are:

$$F_c = k_c e_p + b_c \frac{d}{dt} e_p,$$
$$T_c = k_\theta \alpha a + b_\theta \omega_\theta,$$  \hspace{1cm} (5.2)

for the rotation $e_\theta$ expressed by an angle ($\alpha$) and axis ($a$) of rotation and $\omega_\theta$ denoting the relative angular velocity between the HIT and the VO, $\omega_\theta = \omega_{HIT} - \omega_{VO}$.

### 5.3 Single contact point

A point based approach can be used when modelling the interaction between the tip of a surgical instrument and a virtual soft tissue, which is often applied in Minimally Invasive Surgery. In the following we will explain in detail, how the method of virtual proxy is applied to a single point contact problem. A quasi-static simulation of the proxy point will be used. The contact forces will be generated using a penalty based method. After computing the contact force, we will apply filtering and smoothing techniques in order to avoid feeling force discontinuities at edges of the polygonal object. At last, several friction models, each of which captures different frictional effects, will be described.

#### 5.3.1 Quasi-static proxy point model

The virtual proxy model differentiates between two states of motion: a free space motion and a constrained motion due to active collisions. During a free space motion no force output is generated and the virtual proxy point (VP) follows the haptic interface point (HIP). If a collision occurs, the nearest point to the HIP is found on the surface of the deformable body. We apply the tracing algorithm described in Section 4.4.4 for determining the position of the VP.
To generate the contact force between the tool and the soft body, a linear spring connecting the VP and the HIP is considered. Thus, based on the penetration depth, the external force is obtained:

\[
F_E = k_E (p_{VP} - p_{HIP}),
\]

for \( k_E \) giving the stiffness of the penalty force. The external force \( F_E \) is rendered at the haptic device and also accordingly distributed to the deformable model (see Section 4.4.5).

### 5.3.2 Smoothing the force output

**Filtering the virtual proxy positions**

The actual position of the VP is updated in the haptic loop. The fast update rate of the VP positions promises enhanced stability when rendering stiff contacts. Nevertheless, since the geometry of the deformable body is updated in the slow simulation loop, large discontinuities in the VP position appear after each simulation step. These sudden changes lead to reduced smoothness and stability of the haptic feedback. To lessen these effects, we linearly interpolate between the positions of the VP.

**Force shading paradigm**

When moving with the virtual tool over the edges of the triangular surface mesh, discontinuities of the force magnitude and direction, generated by the virtual proxy model, appear (see Figure 5.3). Without applying force smoothing effects, the user

![Figure 5.3: Force discontinuities due to the use of the virtual proxy paradigm. Large changes in force direction as well as force magnitude are apparent.](image-url)
will feel the edges of the surface triangles and will perceive rather a faceted object instead of a smooth, curved surface.

To reduce the feeling of force discontinuities, characteristic for haptic rendering of polygonal models, we applied the force shading algorithm proposed by Ruspini et al. [1997]. At every step, prior to the VP position adjustments, a plane passing through the current position of the VP is constructed (see Figure 5.4). This plane is denoted as shaded contact plane. It is defined by a normal obtained via linear interpolation of the normals given at the vertices of the triangle in contact. Then, a sub-goal point is defined as the nearest point to the HIP (i.e. the real position of the surgical tool) located on the shaded contact plane. The sub-goal point is used instead of the HIP in the tracing algorithm, when searching for the nearest point on the surface of the deformable body.

![Figure 5.4: Shaded proxy point adjustment. A new sub-goal point is defined to which the nearest point on the triangle surface is searched for. The nearest point found becomes the new proxy point.](image)

It is known (see [Ruspini et al., 1997]), that the force shading algorithm can increase the distance between the HIP and the VP. This implies that energy is introduced into the system artificially. As such, the virtual simulation is not passive any more, which could lead to instability problems. Nevertheless, provided the update rate of the VP positions is high, the added energy is very small and not noticeable by the user. Moreover, the blending of the VP positions as well as the fact that the surface deforms reduces the extent of the parasitic energy. In fact, the stability of the force feedback in our experiments increased by applying the force shading technique.

### 5.3.3 Surface friction

Besides the deformation forces, the friction forces are an important part of the tool-tissue interaction. The computation of friction forces is a necessary component of the interaction force which allows for a controlled manipulation of objects (i.e. lifting up or turning). Without friction the tool will slide off the touched object very easily which would result in an unrealistic slippery feeling.
Coulomb’s friction model can be used to model the basic dry friction characteristics. This model differentiates between two modes: a static and a dynamic friction. During static friction (stiction) a sufficiently high force has to be applied, which will prevent the tool from gliding on the surface. The dynamic friction is applied in case the tool is sliding on the surface. It is applied in opposite direction of the tool motion and damps the velocity of the tool. Both, the maximum stiction as well as the dynamic friction depend on the normal component of the contact force. The dynamic friction is smaller than the stiction.

Considering Coulomb’s friction law, a static friction coefficient $\mu_S$ and a dynamic friction coefficient $\mu_D$ are the material specific parameters which fully describe the friction characteristics. The magnitude of the Coulomb friction force $F_F$ is given by:

$$
F_F \leq \mu_S F_N, \quad \text{if } v = 0,
$$

$$
F_F = \mu_D F_N, \quad \text{otherwise},
$$

(5.4)

where $F_N$ is the normal load and $v$ the velocity of the tool (see also Section 2.2.2 for an introduction to friction models).

\textbf{Friction cone method}

To model the friction force in our environment, we apply a technique know as friction cone method [Harwin and Melder, 2002], which estimates the frictional forces by displacing the VP.

\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{friction_cone.png}
\caption{The friction cone algorithm: a) static friction - the VP position is unchanged; b) dynamic friction - the VP is moved to the edge of the friction cone. Picture taken from [Harwin and Melder, 2002].}
\end{figure}

At every time step, a friction cone oriented perpendicular to the surface is defined at the HIP (see Figure 5.5). The opening angle of the cone is set to $\alpha = 2 \tan^{-1} \mu$, where $\mu$ is set according to the current friction state, which could be either static ($\mu = \mu_S$) or dynamic ($\mu = \mu_D$).
Since the contact force is estimated by the penetration depth of the HIP, the friction cone volume restricts possible positions of the VP to those, for which the tangential force would be smaller than the friction force. Thus if the current VP position is within the friction cone, static friction should be applied. Moreover, since in the static case the friction force should be equal to the tangential component of the interaction force, no adjustment of the VP position is necessary. The new interaction force, adapted for friction, is defined by the distance between the VP position and the HIP (see Equation 5.3).

If the VP lies outside the friction cone, the friction force will not be sufficiently strong to prevent the tool from moving. In this case the VP position has to be adjusted and the dynamic friction force has to be rendered. The new VP position is moved to the boundary of the friction cone, as for this configuration the tangential component of the haptic force will be equal to the friction force. Note, that the friction cone is defined by the dynamic friction coefficient in this case.

While applying the friction cone algorithm, it is necessary to distinguish between a static and a dynamic state of the VP. Based on the state, the corresponding friction coefficient, i.e. static ($\mu_S$) or dynamic ($\mu_D$), is used to define the opening angle of the friction cone. A state transition diagram used for changing the friction conditions is shown in Figure 5.6.

![Figure 5.6: Friction cone state diagram.](image)

The VP is kept in the static state while the VP position is inside the friction cone. It will be changed to the dynamic state, once the VP leaves the friction cone space. Thus, if for a static contact the VP gets outside the friction cone parametrised by $\mu = \mu_S$, the state will be changed to dynamic. The new VP position is then found at the friction cone boundary parametrised by the dynamic friction coefficient $\mu = \mu_D$. On the other side, once for a dynamic contact the VP will stay inside the friction cone parametrised by $\mu = \mu_D$, the state will be changed to static. No correction of the VP position will be taken in this case, since $\mu_S \geq \mu_D$.

The adjustment of the friction cone algorithm to the force shading algorithm, applied in our framework, is straightforward (see Figure 5.7). Given the current position of
the VP, the friction cone algorithm is first applied at the shaded plane. This defines a new position of the sub-goal, used in the force shading algorithm. The new sub-goal is then adjusted according to the friction cone rules. Finally, a new VP position is found by applying the nearest feature tracking algorithm (see Section 4.4.4) to the friction adjusted sub-goal.

**Figure 5.7:** The force shading paradigm adjusted for frictional contact. The friction cone algorithm [Harwin and Melder, 2002] was applied at the shaded contact plane in order to define a new, friction dependent, sub-goal position.

### Extending the friction model

In the following we will extend the friction cone algorithm based on the Coulomb’s friction to more complex friction compensation models. Namely, the modified Dahl’s and the Elasto-plastic friction models, as introduced in [Hayward and Armstrong, 2000] and [Dupont et al., 2000], will be discussed. The new models predict more precisely the friction at low velocities, allowing for presliding effects due to local elastic deformation. The changes proposed in [Hayward and Armstrong, 2000] adapt the Dahl’s friction model to four regimes of friction: modelling stiction, creeping, slipping and sliding. The application of this model to haptic rendering, the necessary parameter identification, as well as a description on the influence of the model parameters on the rendered frictional effects can be found in [Richard, 2000]. The Elasto-plastic model, a modification of the Lu-Gre friction model includes Stribeck’s effect and viscous friction in addition.

We shortly revisit the one dimensional equations describing the latter two friction models (see also Section 2.2.2 for a short introduction). The Dahl friction model is given by:

\[
F_F = \sigma_0 z, \quad \sigma_0 > 0, \quad (5.5)
\]

\[
\dot{z} = \frac{\sigma_0}{F_C} \text{sgn}(\dot{z}) z, \quad (5.6)
\]
where \( z(t) \) specifies the state of elastic strain in the frictional contact, \( x(t) \) the position of the tool, \( \sigma_0 \) the bristle stiffness and \( F_C \) the Coulomb friction (the dynamic friction case is considered here).

The Elasto-plastic friction model is given by:

\[
F_F = \sigma_0 z + \sigma_1 \dot{z} + \sigma_2 \dot{x}, \quad \sigma_0, \sigma_1, \sigma_2 > 0, \quad (5.7)
\]

\[
\dot{z} = \dot{x} \left( 1 - \frac{\sigma_0}{F_{SS}(\dot{x})} \text{sgn}(\dot{x}) \right), \quad (5.8)
\]

where \( \sigma_1 \) and \( \sigma_2 \) are parameters of additional viscous friction, and \( F_{SS} \) is a parameter given by accounting for Stribeck’s effect.

The Dahl’s and the Elasto-plastic friction can be modelled by tracking two points: a moving point belonging to the interaction tool and an adhesion point attached to the body being in contact. The friction force is defined proportional to the strain between the moving point and the adhesion point, as if the two objects were attached by a spring. If a large force is applied, the adhesion point is repositioned to limit the strain of the virtual spring. This corresponds to the idea of the friction cone method where the moving point is defined by the nearest surface point to the HIP and the adhesion point is modelled by the friction adjusted VP. The maximum strain of the virtual spring modelling the friction force is limited by the size of the friction cone. The difference between the Dahl, the Elasto-plastic and the Coulomb based friction cone models is in the adjustment of the VP positions.

Let \( p_{SP} \) denote the position of the nearest surface point to the HIP. Based on [Hayward and Armstrong, 2000] the following update rules at a time step \( t \) are applied when modelling Dahl’s friction:

\[
\begin{align*}
\mathbf{p}'_{VP} &= \begin{cases} 
\mathbf{p}_{SP} - \frac{\mathbf{z}'}{||\mathbf{z}'||} \mathbf{ba} & \text{if } \alpha(||\mathbf{z}'||) \mathbf{z}' > 1, \\
\mathbf{p}_{VP}^{-1} + ||\mathbf{p}_{SP} - \mathbf{p}_{SP}^{-1}|| \alpha(||\mathbf{z}'||) \mathbf{z}' & \text{otherwise,}
\end{cases} \\
\mathbf{z}' &= \mathbf{p}_{SP} - \mathbf{p}_{VP}^{-1}
\end{align*}
\]

(5.9)

where \( \mathbf{z} \) defines the strain of the virtual spring. The function \( \alpha(.) \) is used to model the creeping behaviour of the friction model:

\[
\alpha(z) = \frac{1}{\mathbf{ba} \mathbf{z}^8} \frac{z^8}{\mathbf{ba} \mathbf{z}^8 + \mathbf{z}^8}, \quad 0 \leq \mathbf{ba} \mathbf{z} \leq \mathbf{z}_{\text{stick}}.
\]

(5.10)

Here, the parameters \( \mathbf{ba} \mathbf{z} \) and \( \mathbf{z}_{\text{stick}} \) correspond to the radius of the dynamic friction cone and the static friction cone, measured at the shaded plane. The parameter \( \mathbf{ba} \mathbf{z} \) represents a value, below which the presliding is purely elastic. If the friction strain \( z \) is between \( \mathbf{ba} \mathbf{z} \) and \( \mathbf{z}_{\text{stick}} \), creeping of the adhesion point is observed.
Compensating for Stribeck’s effect, the last equations can be extended to the Elasto-plastic model. First, the function \( \alpha(z) \) is adjusted (see [Dupont et al., 2002]):

\[
\alpha^*(z, v_{SP}) = \frac{1}{2} \left[ 1 + \sin \left( \pi \frac{z - 0.5 (z_{ss}(v_{SP}) + z_{ba})}{z_{ss}(v_{SP}) - z_{ba}} \right) \right],
\]

\[
\alpha(z, v_{SP}) = \begin{cases} 
0, & |z| \leq z_{ba}, \\
\alpha^*(z, v_{SP}), & z_{ba} < |z| < z_{ss}, \\
1, & |z| \geq z_{ss}, 
\end{cases} \tag{5.11}
\]

where \( v_{SP} \) is the velocity of the surface point. The function \( \alpha(\cdot) \) depends on two major values: the break away strain \( z_{ba} \), introduced already for Dahl’s model, and a steady state strain \( z_{ss} \), which is given by the Stribeck curve:

\[
z_{ss} = \frac{1}{\sigma_0} \left( F_D + (F_S - F_D) e^{-\left(\frac{\|v_{SP}\|}{v_S}\right)^2} \right), \tag{5.12}
\]

where \( \sigma_0 \) is the stiffness of the friction force and \( v_S \) is the Stribeck velocity. We use \( \sigma_0 \) equal to the penalty force stiffness \( k_E \) since it defines the contact force, including the friction force. The discrete update rule now changes slightly:

\[
z'^{t} := p_{SP}^{t} - p_{VP}^{t-1}
\]

\[
p_{VP}^{t} = \begin{cases} 
\frac{p_{SP}^{t} - \frac{\sigma_1}{\sigma_0} z^{t}}{\|z^{t}\|} z_{ba}, & |z^{t}| \leq z_{ba}, \\
\frac{p_{SP}^{t} - \frac{\sigma_2}{\sigma_0} v_{SP}}{\|p_{SP}^{t} - p_{VP}^{t-1}\|} \alpha(\|z^{t}\|) z_{ss}^{-1} z^{t}, & |z^{t}| > z_{ss}, \\
\frac{p_{SP}^{t} - \sigma_1}{\sigma_0} z^{t} - \sigma_2 v_{SP}, & |z^{t}| > z_{ss}, \end{cases} \tag{5.13}
\]

Adding the viscosity terms of the Elasto-plastic function (see Equation 5.7) can be also achieved by displacing the VP by \( \Delta z \):

\[
\Delta z = -\frac{\sigma_1}{\sigma_0} z^{t} - \frac{\sigma_2}{\sigma_0} v_{SP}. \tag{5.14}
\]

Here again, the parameter \( \sigma_0 \) is given by the penalty force stiffness \( k_E \). Nevertheless, note that although it is possible to compensate for the viscous friction terms by displacing the VP, it is more precise and practical to add them directly to the tangential force, just before rendering the contact force at the haptic device.

At last we want to point out that the presented algorithms are implicitly dependent on time. Especially the computation of the value \( z \) depends on the sampling rate of the haptic algorithm. Therefore, the perception of the frictional effects, i.e. the stick, slip or the creep behaviour, is influenced by the update rate of the haptic loop.

**Comparison of the friction models**

To compare the different behaviour of the presented models we carried out a simple experiment. We recorded the friction forces of the three presented models, while
sliding with a tool on the top of a square block. The contact was modelled only between the tool-tip of the virtual probe and the surface of the block, which was fixed on one side and deforming during the interaction.

At the beginning, the motion of the probe was recorded at haptic rate. The tool was moved along the x-axis (horizontal), while pushing on the soft block. The captured positions (along the x-axis) are depicted on Figure 5.8 (left). During the test, the recorded input positions have been read in the haptic thread instead of querying the position of the stylus of the haptic device. We run the test with the same input data for each of the three friction models, i.e. the Coulomb, adapted Dahl and the Elasto-plastic model. The corresponding parameters of all the frictional models were kept equal.

![Figure 5.8](image)

**Figure 5.8:** Left: synthesised positions (x-direction) of the VP for different friction models and the input signal. Right: computed frictional forces along the x-axis. On both graphs, the recorded data of each model have been shifted along the vertical axis in order to make the differences of the plots more visible.

During the test, we saved the positions of the VP (the adhesion point) and the friction forces. The plots of the acquired data are shown in Figure 5.8. In the graphs the differences between the friction force responses can be observed. While all the models exhibit the stick and slip behaviour (see the steps in the VP positions or the peaks in the friction force output), the additional friction modes of the extended models are also visible. In Dahl’s model, creeping of the VP can be seen at 3.5 seconds. For the Elasto-plastic friction, the stick-slip effect is reduced in the second half of the plot due to the effect of the velocity dependent terms (compensating for the Strubeck’s effect and viscous friction). This leads to a smooth change of the VP positions and attenuated vibrations of the friction force. On the contrary, using Coulomb’s friction model a frequent change between the stick and the slip states is visible. Presenting this signal to the user may thus lead to a perception of an artificial friction force simulation.
The presented graphs show that additional friction phenomena can be modelled by simple adjustment of the rules for the update of the VP. The extended models allow for more accurate physical behaviour, as described in [Hayward and Armstrong, 2000; Dupont et al., 2002]. Nevertheless, it remains to be determined if the quality of the force feedback is more suitable for the simulation of tool-tissue interactions than the simple friction cone method based on Coulomb’s laws. It has to be remembered, that the choice of the friction model depends on the specific application, i.e. on the frictional parameters of the materials in contact. The extended models, however, provide more degrees of freedom for adjusting the frictional characteristics to a given task. Further analysis and additional psychophysical tests are required to determine which of the models feels most realistic and to find out how sensitive the user is to the additional friction modes.

5.4 Multiple contact points

5.4.1 Quasi-static simulation of the proxy object

Modelling 6-DOF contacts with a rigid tool requires collision detection and force feedback generation for multiple points. The idea of tracking the nearest surface point to a given HIP, i.e. the virtual proxy paradigm, can be easily extended to tracking a set of points defined at the haptic interface tool (HIT). In this sense, several VPs are tracked simultaneously on the surface of the deformable body (see Figure 5.9).

Each of the VPs generates a penalty force \( F_E \) (see Equation 5.3) which is applied on the deformable body as well as on the HIT. As pointed out in [Kim et al., 2002], discontinuities in the number of contacts affect the stability of penalty based simulations. This is due to the total contact stiffness depending on the number of contacts. We therefore average the contact force and torque by the number of VPs being in contact with the deformable body, before rendering on the haptic device. Using a fixed set of points \( P = \{p_1, \ldots, p_n\} \), defined at the HIT, the resulting 6-DOF force feedback is computed by:

\[
\begin{align*}
F_E &= \frac{1}{cp} \sum_{i=1}^{n} F_{E_i} = \frac{k_E}{cp} \sum_{i=1}^{n} (p_{VP_i} - p_i), \\
T_E &= \frac{1}{cp} \sum_{i=1}^{n} (p_i - p_{CM}) \times F_{E_i},
\end{align*}
\]

(5.15)

where \( p_{CM} \) represents the centre of mass of the HIT and \( cp \) denotes the number of points being in contact. The equations provide contact forces only for HIPs which collided with the deformable body. For all others the term \((p_{VP_i} - p_i)\) is zero since the corresponding VP is aligned with the HIP.
The position of each VP is adjusted with respect to the frictional method used (see Section 5.3.3). Therefore, this straightforward extension enables haptic rendering of 6-DOF frictional contact, based on distributed contact points. However, since all feature points are handled individually, the found proxy points at the surface of the deformable body do not preserve the shape of the virtual tool, i.e. straight lines can appear curved and discontinuous (see Figure 5.10). Moreover, when modelling contact with a convex body, the distance between the proxy points tracked at the surface increases as the user pushes the HIT deeper into the object. Also, when the size of the deformable body is small, the user can pass through the object by applying sufficient force.

The main cause of this unrealistic contact modelling is the missing collision detection between the edges of the virtual bodies. To compensate for this undesirable behaviour, the stiffness of the penalty force should be set very high. In that case the surface of the virtual body will be pulled very close to the HIT, providing more precise estimation of the VPs. Nevertheless, to provide stable interaction the stiffness between the haptic device and the virtual environment is limited [Barbagli et al., 2005; Colgate et al., 1995]. It depends on the technical characteristics of the computational hardware and the haptic device (due to limited update rate and strength
of the haptic device). Therefore, it is not possible to completely resolve the problem just by increasing the stiffness of the penalty force computed between the HIT and the nearest surface points.

### 5.4.2 Dynamic simulation of the proxy object

The previously discussed problem caused by setting high stiffness of the contact force can be avoided by using virtual coupling (see Section 5.2), thus decoupling the penalty force computation from the force feedback generation routine. This way we can increase the penalty force stiffness without a major influence on the stability of the haptic feedback. The stiffness will therefore not be limited by the technical parameters of the haptic device. Nevertheless, restrictions have to be applied in order to insure the numerical stability of the simulated virtual environment.

To apply the method of virtual coupling, we will introduce a new virtual tool (VT), to which the haptic interface will be linked by a linear and angular spring and damper elements. The behaviour of the VT in the scene will be modelled dynamically. The contact forces, and thus the collision detection and response algorithms, will be computed with respect to the new VT instead of the HIT (see Figure 5.11).

![Figure 5.11: Collision response with multiple points for the dynamic approach, with applied virtual coupling.](image)

Two types of forces are applied at the dynamic VT: contact and coupling forces. While the penalty based contact force generates the frictional response of the VT to collisions, the coupling force is driving the VT according to the movement of the haptic device. Thus, applying Equations 5.15, 5.1, 5.2 the net force and torque acting at the VT are:

\[
\tilde{\mathbf{F}} = \mathbf{F}_E + \mathbf{F}_c, \\
\tilde{\mathbf{T}} = \mathbf{T}_E + \mathbf{T}_c,
\]

\[\text{(5.16)}\]
with contact forces

\[ F_E = \frac{1}{cp} \sum_{i=1}^{n} F_{E_i} = \frac{k_E}{cp} \sum_{i=1}^{n} (p_{VP_i} - p_{VT_i}), \]

\[ T_E = \frac{1}{cp} \sum_{i=1}^{n} (p_{VT_i} - p_{VT_{CM}}) \times F_{E_i}, \quad (5.17) \]

and coupling forces

\[ (\alpha, a) = q_{HIT} q_{VT}^{-1}, \]

\[ F_c = k_c (p_{HIT} - p_{VT}) + b_c (v_{HIT} - v_{VT}), \]

\[ T_c = k_\theta \alpha a + b_c (\omega_{HIT} - \omega_{VT}). \quad (5.18) \]

The values \( \omega_{HIT} \) and \( \omega_{VT} \) represent the angular velocities of the haptic device and the virtual tool respectively. Finally, the dynamics of the VT is given by the equations of motion of a 6-DOF rigid body:

\[ \dot{p}_{VT} = v_{VT}, \quad \dot{q}_{VT} = \frac{1}{2} \omega_{VT} q_{VT}, \]

\[ m \ddot{v}_{VT} = \tilde{F}, \quad M \ddot{\omega}_{VT} = \tilde{T}. \quad (5.19) \]

Here, we define the state of the tool by the position of its centre of mass \( p_{VT} \), the velocity of the centre of mass \( v_{VT} \), a quaternion \( q_{VT} \) that describes the orientation, and the angular velocity \( \omega_{VT} \). The parameter \( m \) expresses the mass of the object and the matrix \( M \) the inertia matrix of the VT. We would like to note, that we apply the coupling forces and torques at the centre of mass of the VT. This way the positional and the rotational equations are not coupled. We solve the Equations 5.16, 5.17, 5.18, 5.19 in the fast haptic loop applying an explicit numerical integration scheme. For the integration we use a time step equal to the real time delay between two runs of the haptic loop, i.e. 1ms, which allows for a real time simulation of the VT motion.

Since \( \tilde{F} \) and \( \tilde{T} \) include the penalty as well as the coupling forces, the penetration depth of the VT now depends on the penetration force stiffness and the virtual coupling stiffness parameters. If we consider the stiffness of the virtual coupling fixed (i.e. as an internal parameter of the given haptic device), the penetration depth, and thus the correct position of the VPs, is given by the penetration force stiffness.

Finally, the force output rendered on the haptic device is obtained in the haptic loop while solving for the coupling force and torque of the VT. Namely, the force \( -F_c \) and the torque \( -T_c \) are presented to the user. Since the VT is responding to collisions, the contact forces are naturally filtered and transferred to the haptic device using the virtual coupling paradigm.
5.4.3 Summary of the penalty based haptic rendering algorithm

The computation flow of the haptic loop can be summarised as follows.

1. Read positions of the HIT and the VPs

2. Set up a virtual interaction tool (VIT):
   - set VIT = HIT, if quasi-static proxy is used
   - set VIT = VT, if dynamic proxy is used

3. Find the nearest surface points for the VIT;
   For each surface point do:
   - Define goal-point by finding the nearest point on the corresponding shaded plane
   - Apply friction cone method to the goal-point
   - Find nearest surface point to the goal-point using the tracking algorithm introduced in Section 4.4.4

4. Compute the contact force based on the distance between the VIT and the nearest surface points (see Equations 5.15, 5.17 for the HIT or the VT respectively)

5. Store the current contact force for the simulation loop (the contact force will be requested by the simulation thread when necessary)

6. If a quasi-static proxy is used:
   - Filter the positions of the nearest surface points
   - Update the position of each VP to the filtered position of the corresponding nearest surface point

otherwise, if a dynamic proxy is used:
   - Add the contact force to the external force applied on the VT
   - Compute the coupling force between HIT and VT
   - Add the coupling force to the VT
   - Update the dynamic state of the VT (use explicit or implicit integration with the time step set to 1 ms)

7. Set the haptic force:
   - Use contact force applied at the VIT, if a quasi-static proxy is used
   - Use coupling force between HIT and VT if a dynamic proxy is used
5.4.4 Setting the parameters of the penalty based haptic rendering algorithm

Setting the parameters of the penalty force for the quasi-static simulation of the VPs must be done in consideration with the stiffness of the deformable body as well as the mechanical properties of the haptic device. It is limited by the refresh rate of the simulation as well as the haptic loop. Currently, no automatic method for setting the best values of the contact force stiffness is known for complex haptic environments. Therefore, we set all necessary parameters of the quasi-static haptic rendering algorithm empirically.

The dynamic approach introduces additional parameters, which affect the stability of the simulation. In general, we first set the coupling stiffness, damping and the mass of the VT for a free space interaction such that no delays are apparent to the user. Note, that the integration time step is set by default to $1 \text{ ms}$, which represents the update frequency of the haptic loop.

While we tend to set the coupling stiffness as high as possible, the VP mass is set to a low value to minimise the perception of the inertia forces. Although a technique of simulating a proxy point with zero mass dynamically was proposed by Niemeyer and Mitra [2004], we prefer to include the mass of the proxy point into the dynamic equations. The rational of this is that since the proxy object should simulate a virtual tool in the scene, the mass of the real tool should be used in the dynamic equations. We therefore use the real inertia parameters of the VT for the 6-DOF simulation of a rigid tool.

Note that, due to the use of an explicit integration scheme when solving for the equation of motion of the 6-DOF VT (see Equation 5.19), the application may get instable for small values of mass and momentum of inertia. To increase the numerical stability, an implicit integration scheme, as presented in [Otaduy and Lin, 2006], can be applied instead. Nevertheless, the stability is dependent also on the integration time step. Since we solve for the motion of the rigid tool in the haptic loop, the haptic refresh rate provides a sufficiently small time step for stable computation while ensuring real time simulation of the VT.

After the coupling force parameters are set, the penalty force stiffness is adjusted to a high value, in order to make the penetration of the tool not too visible.

5.5 Constraint based formulation of the contact forces

In this section we will extend the dynamic 6-DOF force feedback algorithm presented in Section 5.4.2 by exchanging the penalty based approach of the contact force computation with a constraint based method.
The penalty based simulation exhibits several problems which lead to reduced smoothness of the contact forces and unrealistic contact force estimation, especially for large penetrations. These are mainly caused by the rough approximations of the penetration volume. Recall that once the rigid body enters into the deformable object, the collision points at the surface are not necessarily preserving the shape of the rigid tool. Furthermore, discontinuities appear also due to the dependence of the force stiffness on the number of contacts.

The constraint based force computation overcomes these issues. It determines contact forces which restrict the motion of the VT and the deformable body in order to satisfy Signorini’s contact law (Equation 2.11). Therefore, the nonpenetration constraints are handled accurately, which provides more exact force response. Moreover, the constraint based approach also simplifies the parameter setting. For the formulation of the contact constraints only the material properties of the collided objects and several simulation specific parameters (i.e. the time step) have to be considered. No contact force stiffness and damping has to be defined, as in penetration based methods. Only the specific values for the virtual coupling paradigm have to be set up in addition to the material parameters, in order to ensure a stable haptic interaction.

We based our contact model on the developments of Otaduy and Gross [2007], who used velocity based constraints to compute the collision forces. The novelty of their model is in using the dynamically consistent inverse of the contact Jacobians for a fast contact force update within a multirate haptic simulation. First, the velocities at the constraints are defined based on the generalised contact Jacobians and the velocities of the rigid and deformable virtual solids. Afterwards an inverse of the contact Jacobians, dynamically consistent with the constraints, is defined. Finally, these are used to approximate the contact forces in the fast haptic thread. As a result, a highly transparent rendering is achieved.

In the approach of Otaduy and Gross [2007] frictionless equality constraints are applied. We have extended the frictionless contact force formulation to inequality constraints based on the Signorini’s contact model and adapted the presented approach to compensate for frictional effects. In the following sections we will describe the formulation of the contact constraints, computation of contact forces and adaptations necessary to derive a multirate architecture of the haptic simulator.

### 5.5.1 General overview

The constraint based haptic rendering differs from the previous approach also in the computation flow of the contact forces (see Section 5.4.3). Otaduy and Gross [2007] present a novel approach to the multirate haptic rendering. A general overview of the multirate architecture is depicted in Figure 5.12.
Figure 5.12: Multirate rendering algorithm as proposed in [Otaduy and Gross, 2007].

The idea is in splitting the haptic interaction into two components, a simulation and a haptic one, where each of them will simulate a separate interaction of a rigid tool. In the main system, the interaction of a visual tool, coupled to the haptic device, and the deformable body is simulated. The collision detection and the constraint based collision response is applied between the visual tool and the soft body. While solving for the contact forces, necessary parameters for a linearised contact model are updated. No haptic feedback is generated in this thread. This component is running in the simulation thread at a slow update rate. In the second system, the simulation of a haptic tool is performed. Here, the force feedback is generated based on the linearised contact model (for details see Section 5.5.4) with internal parameters precomputed in the simulation thread. This component is running in the fast haptic thread at 1 kHz.

The control flow of the constraint based haptic rendering algorithm is given below. Simulation loop:

1. Read current state of the haptic interface
2. Apply coupling forces to the visual tool
3. Solve collision free update of the visual tool and the deformable body (see Section 5.5.2)
4. Detect the collisions and construct the contact constrains (see Section 5.5.3)
5. Solve for the contact forces and update accordingly the post-collision state (see Appendix C)
6. Update the parameters of the linear contact model (see Section 5.5.4)

Haptic loop:
1. Read current state of the haptic interface

2. Apply coupling force to the haptic tool

3. Solve collision free update of the haptic tool

4. Apply the collision forces using the linear contact model (see Section 5.5.4)

5. Update the post-collision state of the haptic tool

6. Compute the coupling forces to be rendered on the haptic device

### 5.5.2 Contact-free dynamics of the virtual bodies

In this section we describe how to decouple the collision-free update from collision response. Since velocity constraints will be imposed on the dynamic simulation of the VT as well as the nodes of the deformable body, we will only consider the velocity update of the virtual objects.

To simplify the notation we will refer in the following to the state variables of the rigid body by indices $r$ and $\omega$, for the linear and the angular variables respectively, while for the nodes of the deformable body we will use an index $d$.

#### Simulation of the rigid tool

The update of the rigid body is given by the Equations 5.19. Separating the collision forces and torques from the others ($F$ and $T$) the net force and torque (see Equations 5.16) can be given:

$$
\tilde{F} = F + J_r^T \lambda + H_r^T \zeta,
$$

$$
\tilde{T} = T + J_\omega^T \lambda + H_\omega^T \zeta,
$$

(5.20)

while splitting the contact forces to their normal and tangential components, denoted by $\lambda$ and $\zeta$ respectively. The direction of the normal component for each contact point is perpendicular to the contact surface. It is given by the outer normal of the deformable body at the contact point. The normal vectors for each contact point are included in the matrices $J_r$ and $J_\omega$. Similarly, the matrices $H_r$ and $H_\omega$ contain the information of the tangential contact space for each collision point. The vectors $\lambda$ and $\zeta$ thus denote the collision forces expressed in the contact space. Note, that the matrices $H_r$ and $H_\omega$ as well as the vector $\zeta$ form a two dimensional space for each contact point. The formulation of the contact force matrices is described in detail in the following section.
Given this separation of forces at every time step, we can divide the update of velocities to a collision-free part \( v^r - r \) and a collision impulse \( \delta v_r \) (and similar for the angular velocities). Then, accounting for the time-discretisation of the equations of motion, the linear and the angular velocity update can be solved by the following linear systems:

\[
\begin{align*}
\hat{M}_r (v^r - r + \delta v_r) &= \Delta t \hat{F} + J^T_r \tilde{\lambda} + H^T_r \tilde{\zeta}, \\
\hat{M}_\omega (\omega^r + \delta \omega_r) &= \Delta t \hat{T} + J^T_\omega \tilde{\lambda} + H^T_\omega \tilde{\zeta},
\end{align*}
\]

(5.21)

\[
\begin{align*}
v^r &= \Delta t \hat{M}_r^{-1} \hat{F}, & \delta v_r &= \hat{M}_r^{-1} (J^T_r \tilde{\lambda} + H^T_r \tilde{\zeta}), \\
\omega^r &= \Delta t \hat{M}_\omega^{-1} \hat{T}, & \delta \omega_r &= \hat{M}_\omega^{-1} (J^T_\omega \tilde{\lambda} + H^T_\omega \tilde{\zeta}),
\end{align*}
\]

(5.22)

with \( \tilde{\lambda} = \Delta t \lambda \) and \( \tilde{\zeta} = \Delta t \zeta \) denoting the collision impulses\(^1\) applied during one simulation time step \( \Delta t \).

To obtain the discrete time matrices \( \hat{M}_r \) and \( \hat{M}_\omega \), and the discrete time force vectors \( \hat{F} \) and \( \hat{T} \), we have used an implicit backward Euler scheme, as this allows for stable simulation of the rigid tool in the slow simulation thread. We can write:

\[
\begin{align*}
\hat{M}_r &= m I - \Delta t \frac{\partial F}{\partial v_r} - \Delta t^2 \frac{\partial F}{\partial p_r}, \\
\hat{F} &= F + \left( \frac{m}{\Delta t} I - \frac{\partial F}{\partial v_r} \right) v_r, \\
\hat{M}_\omega &= M - \Delta t \frac{\partial T}{\partial \omega_r} - \Delta t^2 \frac{\partial T}{\partial q_r} \hat{Q}, \\
\hat{Q} &= (I - \Delta t \Omega)^{-1} Q, \\
\hat{T} &= T + \left( \frac{1}{\Delta t} M - \frac{\partial T}{\partial \omega_r} \right) \omega_r,
\end{align*}
\]

(5.23)

with \( \Omega \) and \( Q \) representing the quaternion product \( \dot{q}_r = \frac{1}{2} \omega_r q_r = \Omega q_r = Q \omega_r \) as matrix-vector multiplication. Currently only the coupling forces are applied in addition to the collision forces. The detailed computation of the coupling force and torque Jacobians can be found in [Ostaduy and Lin, 2006].

**Simulation of the deformable objects**

The dynamic equations of motion for the deformable body are described by Equation 4.18. We divide the external forces to contact forces \( J^T_\lambda \lambda, H^T_\zeta \zeta \) and others \( F_E \)

\(^1\)We will refer to collision impulses instead of collision forces to simplify the further notation.
(i.e. the gravitational force), similar as done for the rigid body. After time discretisation of the motion equations we obtain the collision-free velocity update $\mathbf{v}_{d}^-$ and a collision impulse $\delta \mathbf{v}_{d}$:

\[
\hat{\mathbf{M}}_d \left( \mathbf{v}_{d}^- + \delta \mathbf{v}_{d} \right) = \Delta t \hat{\mathbf{F}}_d + \mathbf{J}_d^T \hat{\lambda} + \mathbf{H}_d^T \hat{\zeta}, \\
\mathbf{v}_{d}^- = \Delta t \hat{\mathbf{M}}_d^{-1} \hat{\mathbf{F}}_d, \quad \delta \mathbf{v}_{d} = \hat{\mathbf{M}}_d^{-1} \left( \mathbf{J}_d^T \hat{\lambda} + \mathbf{H}_d^T \hat{\zeta} \right),
\]

(5.24)

for the contact impulses $\hat{\lambda} = \Delta t \lambda$, $\hat{\zeta} = \Delta t \zeta$.

The formulation of the discrete time matrix $\hat{\mathbf{M}}_d$ and the discrete time force vector $\hat{\mathbf{F}}_d$, for several integration schemes, is given in Section 4.4.3. We would like to note that, although the formulations in Section 4.4.3 are specific for our MSS, the above formulation can be obtained for any other deformation model that supports local linearisation.

### 5.5.3 Contact handling

In this section we describe the formulation of constraints and the computation of contact forces as it is running in the simulation thread. The defined constraints will assemble a linear relation between the velocities of the virtual bodies at the contact points and the collision impulses.

**Frictionless contact**

Given the state of the tool and the deformable body after the collision-free update, we perform a collision detection test (see Section 4.4.4) to identify colliding primitives. Considering a collision point $x$ at the surface of the deformable body with the contact normal $\mathbf{n}$ (surface normal of the soft body) we define a relative velocity $w_n$ at the collision point in the normal direction as:

\[
w_n := \mathbf{n}^T \left( \mathbf{v}_r + \omega_r \times (\mathbf{x} - \mathbf{p}_r) - \mathbf{v}_x \right).
\]

(5.25)

Remember that we detect collisions of points defined at the rigid tool with the surface of the soft body. Therefore, in general every collision occurs at some face of the deformable object. To relate the relative velocity to the nodes we use barycentric coordinates of the collision point as in [Duriez et al., 2006]. Let $\mathbf{p}_d^1$, $\mathbf{p}_d^2$ and $\mathbf{p}_d^3$ be the vertices of the surface triangle in contact. For barycentric coordinates $[\alpha, \beta, \gamma]$ of the collision point $x$ we have:

\[
x = \mathbf{N}(x) \begin{bmatrix} \mathbf{p}_d^1 \\ \mathbf{p}_d^2 \\ \mathbf{p}_d^3 \\ \mathbf{p}_d^4 \end{bmatrix}, \quad \mathbf{N}(x) = \begin{bmatrix} \alpha \mathbf{I} \\ \beta \mathbf{I} \\ \gamma \mathbf{I} \end{bmatrix}.
\]

(5.26)
Equivalent relation holds also for velocities $v_x$, $v^1_d$, $v^2_d$ and $v^3_d$.

Now, Equation 5.25 can be rewritten:

$$n^T \left( v_r + \omega_r \times (x - p_r) - N(x) \begin{bmatrix} v^1_d \\ v^2_d \\ v^3_d \end{bmatrix} \right) = w_n,$$

$$j_r v_r + j_\omega \omega_r + j_d \begin{bmatrix} v^1_d \\ v^2_d \\ v^3_d \end{bmatrix} = w_n,$$

$$j_r = n^T, \quad j_\omega = -n^T (x - p_r)^*, \quad j_d = -n^T N(x), \quad (5.27)$$

where $(x - p_r)^*$ represents a cross product in matrix-vector product form.

Assembling all contacts, and dividing the velocities to the collision-free and the collision impulse update, the relative velocities can be expressed in matrix notation:

$$J_r \delta v_r + J_\omega \delta \omega_r + J_d \delta v_d - b_n = w_n,$$

$$b_n = -J_r v_r^- - J_\omega \omega_r^- - J_d v_d^-,$$  \hspace{1cm} (5.28)

for $b_n$ denoting the collision-free relative velocity at the contact points. Substituting the collision impulse induced velocity changes $\delta v_r$, $\delta \omega_r$ and $\delta v_d$ (Equations 5.22 and 5.24 with the tangential component considered zero: $\tilde{\zeta} = 0$) into Equation 5.28, we obtain a linear relation between the relative velocities and contact impulses:

$$A \tilde{\lambda} - b_n = w_n,$$  \hspace{1cm} (5.29)

$$A = J_r \hat{M}_r^{-1} J_r^T + J_\omega \hat{M}_\omega^{-1} J_\omega^T + J_d \hat{M}_d^{-1} J_d^T.$$

The matrix $A$ represents a Delassus operator, which models the behaviour of the solids at the contact points [Renouf and Acary, 2006; Duriez et al., 2006].

Finally, accounting for the nonpenetration constraints yields the following velocity Signorini conditions:

$$w_n \geq 0, \quad \tilde{\lambda} \geq 0, \quad w_n \perp \tilde{\lambda}.$$  \hspace{1cm} (5.30)

The first inequality constraint enforces $w_n \geq 0$ for each collision point $i$, meaning that the two objects currently being in contact can either remain in contact (with zero relative velocity) or move apart. The second condition states that we account only for repulsive forces. In case an equality sign holds for the relative velocity, the contact forces are active. Once the bodies are moving apart, the contact forces vanish. This is expressed by the last term.

The contact problem defined in Equation 5.29 together with constraints given in Equation 5.30 represents a linear complementarity problem (LCP). In [Cottle et al., 1992] several methods for solving the LCP can be found. We applied projected Gauss-Seidel solver, which will be explained in detail for the frictional contact problem in Appendix C.
5.5. Constraint based formulation of the contact forces

Frictional contact

Similar as done for the normal direction in previous section, we define the tangential component of the relative velocity at the collision point. Let the tangential space at the collision point $x$ be spanned by vectors $\{t_1, t_2\}$. Given $\tau = [t_1, t_2]$, and adapting the conditions given for the normal direction (Equation 5.27), we can describe the relative tangential velocity $w_\tau$ as:

$$
\tau^T \begin{pmatrix} v_r + \omega_r \times (x - p_r) - N(x) \end{pmatrix} = w_\tau,
$$

$$
h_r v_r + h_\omega \omega_r + h_d \begin{pmatrix} v^1_d \\ v^2_d \\ v^3_d \end{pmatrix} = w_\tau,
$$

$$
h_r = \tau^T, \quad h_\omega = -\tau^T (x - p_r)^*, \quad h_d = -\tau^T N(x). \quad (5.31)
$$

Note that the relative tangential velocity $w_\tau$ is now a 2D vector and the contact Jacobians $h_r$, $h_\omega$ and $h_d$ are represented by $2 \times 3$ matrices. The equivalent matrix notation for all contacts reads:

$$
H_r \delta v_r + H_\omega \delta \omega_r + H_d \delta v_d - b_\tau = w_\tau,
$$

$$
b_\tau = -H_r v_r - H_\omega \omega_r - H_d v_d, \quad (5.32)
$$

Applying the relations of the collision induced velocities (Equations 5.22 and 5.24), the linear relation between the relative velocities and contact impulses for a frictional contact problem can be expressed as:

$$
\begin{pmatrix} A_{JJ} & A_{JH} \\ A_{HJ} & A_{HH} \end{pmatrix} \begin{pmatrix} \tilde{\lambda} \\ \tilde{\zeta} \end{pmatrix} - \begin{pmatrix} b_n \\ b_\tau \end{pmatrix} = \begin{pmatrix} w_n \\ w_\tau \end{pmatrix}, \quad (5.33)
$$

for the corresponding submatrices given by:

$$
A_{JJ} = J_r \hat{M}_r^{-1} J^T_r + J_\omega \hat{M}_\omega^{-1} J^T_\omega + J_d \hat{M}_d^{-1} J^T_d,
$$

$$
A_{JH} = J_r \hat{M}_r^{-1} H^T_r + J_\omega \hat{M}_\omega^{-1} H^T_\omega + J_d \hat{M}_d^{-1} H^T_d,
$$

$$
A_{HJ} = H_r \hat{M}_r^{-1} J^T_r + H_\omega \hat{M}_\omega^{-1} J^T_\omega + H_d \hat{M}_d^{-1} J^T_d,
$$

$$
A_{HH} = H_r \hat{M}_r^{-1} H^T_r + H_\omega \hat{M}_\omega^{-1} H^T_\omega + H_d \hat{M}_d^{-1} H^T_d.
$$

The constraints for the tangential component of the relative velocities are given by the chosen friction model. We applied Coulomb’s model with static and dynamic friction mode. It suggests that the frictional force is dependent on the normal component of the contact force. For one contact point and the static friction case we have:

$$
\|w_\tau\| = 0 \quad \Rightarrow \quad \|\tilde{\zeta}\| \leq \mu_s \tilde{\lambda}. \quad (5.34)
$$
If dynamic friction applies we get:

$$\|w_\tau\| > 0 \Rightarrow \tilde{\zeta} = -\mu_D \tilde{\lambda} \frac{w_\tau}{\|w_\tau\|},$$

(5.35)

for $\mu_S \geq \mu_D$ describing the static ($\mu_S$) and the dynamic ($\mu_D$) friction coefficients. The term $\frac{w_\tau}{\|w_\tau\|}$ denotes the direction of the relative velocity at the given contact point. It is unknown a priori as it depends on the values of all collision forces. Nevertheless, an approximate direction will be computed while solving for the contact forces (see Appendix C).

Summarising the Equations 5.33, 5.34 and 5.35 we can formulate the frictional contact problem (FCP):

$$A\tilde{\lambda}^* - b = w,$$
$$w_n \geq 0, \quad \tilde{\lambda} \geq 0, \quad \forall i : \mu_s \tilde{\lambda}_i - \|\tilde{\zeta}_i\| \geq 0,$$
$$w_n \perp \tilde{\lambda}, \quad \forall i : w_n \perp \|\tilde{\zeta}_i\|, \quad \forall i : \|w_\tau\| \perp \tilde{\zeta}_i - \mu_D \tilde{\lambda}_i \frac{w_\tau}{\|w_\tau\|} \|w_\tau\|,$$

(5.36)

for $A, \tilde{\lambda}^*, b$ and $w$ set correspondingly to Equation 5.33 and the relations involving the index $i$ considered for each contact point separately.

Note that the frictional contact problem is not a LCP anymore. This is due to adding the nonlinear complementarity conditions of the 3D friction model (Equations 5.34 and 5.35). Approximating the Coulomb friction cone with $n$-sided pyramids is a well-known approach to linearise the frictional conditions [Stewart and Trinkle, 1996; Sauer and Schömer, 1998; Renouf et al., 2005]. Instead we decided to use the full friction cone, similar to Renouf and Acary [2006] and Duriez et al. [2006]. We apply an iterative Gauss-Seidel block splitting method to solve the constraint problem contact by contact. At each iteration we solve a local FCP considering the contribution of other contact points fixed. For the single point contact we apply a projected Gauss-Seidel method. The details of the iterative solver can be found in Appendix C. After we obtain the collision impulses, the state of the deformable body and the visual tool are updated according to Equations 5.22 and 5.24.

5.5.4 Linear contact model

In the haptic thread we evaluate the discrete-time force $\hat{F}$ and torque $\hat{T}$ which depend on the user’s motion transferred through virtual coupling. To estimate the contact forces, in [Otaduy and Gross, 2007] the authors linearise the collision
impulses in the contact space. The contact forces are then evaluated according to the other external forces ($\hat{\mathbf{F}}$ and $\hat{\mathbf{T}}$), i.e. the coupling forces, in the haptic loop by:

\[
\mathbf{F}_{E,H} = \mathbf{F}_{E,0} + \frac{\partial \mathbf{F}_E}{\partial \mathbf{F}} \hat{\mathbf{F}} + \frac{\partial \mathbf{F}_E}{\partial \mathbf{T}} \hat{\mathbf{T}},
\]
\[
\mathbf{T}_{E,H} = \mathbf{T}_{E,0} + \frac{\partial \mathbf{T}_E}{\partial \mathbf{F}} \hat{\mathbf{F}} + \frac{\partial \mathbf{T}_E}{\partial \mathbf{T}} \hat{\mathbf{T}}.
\] (5.37)

Here, the matrices $\frac{\partial \mathbf{F}_E}{\partial \mathbf{F}}$, $\frac{\partial \mathbf{F}_E}{\partial \mathbf{T}}$, $\frac{\partial \mathbf{T}_E}{\partial \mathbf{F}}$ and $\frac{\partial \mathbf{T}_E}{\partial \mathbf{T}}$ form the dynamically-consistent inverse of the contact Jacobians. The calculation of the linear model Jacobians is performed in the simulation loop.

Recall the definition of the contact forces and torques (Equation 5.20):

\[
\mathbf{F}_E = \mathbf{J}_T^T \lambda + \mathbf{H}_T^T \zeta,
\]
\[
\mathbf{T}_E = \mathbf{J}_T^T \omega + \mathbf{H}_T^T \zeta.
\] (5.38)

Using Equation 5.33 one can relate the collision impulses $\tilde{\lambda}$ and $\tilde{\zeta}$ to the collision free velocities $\mathbf{v}_r$ and $\omega_r$ which depend solely on the discrete-time forces $\hat{\mathbf{F}}$ and $\hat{\mathbf{T}}$.

To obtain the discrete time contact Jacobians we split the contacts to three groups: sticking $(s)$, slipping $(p)$ and breaking $(b)$. On the sticking contacts static friction is considered, while on the slipping contacts dynamic friction is applied. The breaking contacts represent collision points which, after the applied contact forces, are moving apart and thus no contact force is applied at them. According to the complementarity conditions in Equation 5.36 the following holds for contact $i$:

\[
\begin{align*}
\mathbf{w}_n &= 0, \quad \tilde{\lambda}_i \geq 0 \quad \{ \text{i sticking} \} \\
\mathbf{w}_\tau &= 0, \quad \| \tilde{\zeta}_i \| \geq 0 \quad \{ \text{i slipping} \} \\
\mathbf{w}_n &= 0, \quad \tilde{\lambda}_i \geq 0, \quad \| \mathbf{w}_\tau \| > 0, \quad \tilde{\zeta}_i = -\mu_D \tilde{\lambda}_i \frac{\mathbf{w}_\tau}{\| \mathbf{w}_\tau \|} \quad \{ \text{i slipping} \} \\
\mathbf{w}_n &\geq 0, \quad \tilde{\lambda}_i = 0, \quad \| \mathbf{w}_\tau \| \geq 0, \quad \tilde{\zeta}_i = 0 \quad \{ \text{i breaking} \}
\end{align*}
\] (5.39)

While the sticking and the slipping contacts represent active constraints for which a linear relation can be obtained using Equation 5.33, the breaking constraints can be eliminated since for them the collision forces are zero.

Rearranging $\tilde{\lambda}^* = [\tilde{\lambda}_s^*, \tilde{\lambda}_p^*, \tilde{\lambda}_b^*]$ for components of sticking, slipping and breaking contacts, respectively, we can express the Equation 5.33 in a form:

\[
\begin{pmatrix}
\mathbf{A}_{ss} & \mathbf{A}_{sp} & \mathbf{A}_{sb} \\
\mathbf{A}_{ps} & \mathbf{A}_{pp} & \mathbf{A}_{pb} \\
\mathbf{A}_{bs} & \mathbf{A}_{bp} & \mathbf{A}_{bb}
\end{pmatrix}
\begin{pmatrix}
\tilde{\lambda}_s^* \\
\tilde{\lambda}_p^* \\
\tilde{\lambda}_b^*
\end{pmatrix} -
\begin{pmatrix}
\mathbf{b}_s \\
\mathbf{b}_p \\
\mathbf{b}_b
\end{pmatrix} =
\begin{pmatrix}
\mathbf{0} \\
\mathbf{w}_d \\
\mathbf{w}_v
\end{pmatrix}.
\]
Furthermore, emphasising the normal \((n)\) and the tangential \((\tau)\) directions of the slipping contacts and removing the equations for the breaking contacts which do not contribute to the collision impacts \((\tilde{\lambda}_b = 0\) according to Equation 5.39) we get:

\[
\begin{pmatrix}
A_{ss} & A_{sn} & A_{st} \\
A_{ns} & A_{nn} & A_{nt} \\
A_{ts} & A_{tn} & A_{tt}
\end{pmatrix}
\begin{pmatrix}
\tilde{\lambda}_s^* \\
\tilde{\lambda}_n^* \\
\tilde{\lambda}_\tau^*
\end{pmatrix}
- \begin{pmatrix}
b_s \\
b_n \\
b_\tau
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
w_\tau
\end{pmatrix},
\]

(5.40)

for \(\tilde{\lambda}_n^* = \tilde{\lambda}_p\) and \(\tilde{\lambda}_\tau^* = \tilde{\zeta}_p\). Recall the Equation 5.35 which applies for slipping contacts. The magnitude of the tangential collision impacts for slipping contacts is given by the normal component \(\|\tilde{\zeta}_i\| = \mu_D \tilde{\lambda}_i\), while the direction is opposite to the tangential relative velocity \(\tilde{\zeta}_i = -\frac{w_\tau}{\|\tilde{\zeta}_i\|}\). Nevertheless, making an assumption that the direction of the friction force will stay constant during one simulation step, the direction obtained when solving the FCP can be used. Therefore, for the two components \(\tilde{\zeta}_i^1\) and \(\tilde{\zeta}_i^2\) of the tangential contact impulses \(\tilde{\zeta}_i = [\tilde{\zeta}_i^1, \tilde{\zeta}_i^2]\) we can write:

\[
\begin{align*}
\tilde{\zeta}_i^1 &= c_i^1 \tilde{\lambda}_i = \mu_D \tilde{\lambda}_i \frac{\tilde{\zeta}_i^1}{\|\tilde{\zeta}_i\|}, \\
\tilde{\zeta}_i^2 &= c_i^2 \tilde{\lambda}_i = \mu_D \tilde{\lambda}_i \frac{\tilde{\zeta}_i^2}{\|\tilde{\zeta}_i\|}.
\end{align*}
\]

These equations express the projection of the friction force on the full friction cone. Here, the terms \(\frac{\tilde{\zeta}_i^1}{\|\tilde{\zeta}_i\|}\) and \(\frac{\tilde{\zeta}_i^2}{\|\tilde{\zeta}_i\|}\) on the right hand side are considered constant according to the assumption we made. Equivalently, we can stack all slipping contacts together:

\[
\tilde{\zeta}_p = C\tilde{\lambda}_p,
\]

(5.41)

for the matrix \(C\) encapsulating the dynamic friction coefficient and the slipping direction for all slipping contacts.

Inserting Equation 5.41 into Equation 5.40 we obtain a linear relation between the collision impulses and the collision free velocities for active contacts:

\[
\begin{pmatrix}
A_{ss} & A_{sn} + CA_{st} \\
A_{ns} & A_{nn} + CA_{nt}
\end{pmatrix}
\begin{pmatrix}
\tilde{\lambda}_s^* \\
\tilde{\lambda}_n^*
\end{pmatrix}
- \begin{pmatrix}
b_s \\
b_n
\end{pmatrix}
= \begin{pmatrix}
0 \\
0
\end{pmatrix}.
\]

(5.42)

In fact, we could have derived the above relation following the construction of the FCP. Considering only the constraints for sticking contacts and the normal direction of the slipping contacts (Equations 5.27, 5.31), and accounting for the dynamic
friction impulses \( \tilde{\zeta}_p = C \tilde{\lambda}_p \) in the collision update of the velocities (Equations 5.22 and 5.24) would lead to the following relations:

\[
\begin{align*}
\tilde{A} &= \begin{pmatrix} A_{ss} & A_{sn} + CA_{st} \\ A_{ns} & A_{nn} + CA_{nt} \end{pmatrix} = G_r \tilde{M}_r^{-1} \tilde{G}_r + G_w \tilde{M}_w^{-1} \tilde{G}_w + G_d \tilde{M}_d^{-1} \tilde{G}_d, \\
\tilde{G}_r &= \begin{pmatrix} J_{r,s} \\ H_{r,s} \\ J_{r,p} + CH_{r,p} \end{pmatrix}, \quad \tilde{G}_w = \begin{pmatrix} J_{w,s} \\ H_{w,s} \\ J_{w,p} + CH_{w,p} \end{pmatrix}, \quad \tilde{G}_d = \begin{pmatrix} J_{d,s} \\ H_{d,s} \end{pmatrix},
\end{align*}
\]

for the sticking \((s)\) and the slipping \((p)\) contacts. Nevertheless, adapting the Equation 5.33 directly gives a better insight into the necessary computations which have to be performed on the system matrix \( \tilde{A} \) constructed before the linearisation of the contact model.

Using Equations 5.42, 5.38 and 5.43 the contact forces which will be applied on the haptic tool are given by:

\[
\begin{align*}
F_{E,H} &= \frac{1}{\Delta t} \tilde{G}_r \tilde{A}^{-1} \begin{pmatrix} b_s \\ b_n \end{pmatrix}, \\
T_{E,H} &= \frac{1}{\Delta t} \tilde{G}_w \tilde{A}^{-1} \begin{pmatrix} b_s \\ b_n \end{pmatrix},
\end{align*}
\]

where \( b_s \) and \( b_n \) are given in Equations 5.28 and 5.32. It is now straightforward to compute the Jacobians of the contact forces according to Equation 5.37. Substituting the collision-free update of the rigid body velocities (Equation 5.22) in terms of \( \hat{F} \) and \( \hat{T} \), we obtain the necessary expressions for our linear contact model, which will allow us to estimate contact forces from external forces in the haptic thread:

\[
\begin{align*}
F_{E,0} &= -\frac{1}{\Delta t} \tilde{G}_r \tilde{A}^{-1} G_d \hat{v}_d, \\
\frac{\partial F}{\partial \hat{F}} &= -\tilde{G}_r \tilde{A}^{-1} G_r \tilde{\hat{M}}_r^{-1}, \\
\frac{\partial T}{\partial \hat{F}} &= -\tilde{G}_r \tilde{A}^{-1} G_r \tilde{\hat{M}}_r^{-1}, \\
T_{E,0} &= -\frac{\partial F}{\partial \hat{T}} = -\tilde{G}_w \tilde{A}^{-1} G_w \hat{\hat{M}}_w^{-1}, \\
\frac{\partial T}{\partial \hat{T}} &= -\tilde{G}_w \tilde{A}^{-1} G_w \hat{\hat{M}}_w^{-1}.
\end{align*}
\]

Note that, in the expressions above, \( \Delta t \) is the time step of the simulation thread.

Obtaining the linear model requires solving six linear systems, \( \tilde{A}^{-1} \tilde{G}_r \) and \( \tilde{A}^{-1} \tilde{G}_w \).

Here we would like to point out, that the matrix \( \tilde{A} \) is often badly conditioned. This is due to the transfer of the contact constraints from the collision points to the nodes of the deformable body. For multiple contact points the surface triangles being in contact often share their vertices, moreover, several collision points can
occur inside of one triangle. Thus, multiple constraints can be imposed on a single node. This results in low conditionality of the matrix $\tilde{A}$, which leads to numerical instabilities when solving the linear systems. Therefore, we compute the pseudo-inverse of the matrix $\tilde{A}$ using a singular value decomposition (SVD) while neglecting the minor singular values. Although this computation presents a potential overhead of our algorithm, in practice the computation time is negligible in comparison to the runtime of the FCP solver. Nevertheless, this calculation is performed only in the slow simulation thread. The evaluation speed of the contact forces in the haptic thread is thus not influenced by the complexity of the matrix $\tilde{A}$. In fact, and as pointed out in [Otaduy and Gross, 2007], the computation of the collision forces requires only four matrix-vector multiplications and four vector additions which allows for very high refresh rates.

### 5.5.5 Limitations

The presented constraint based algorithm employs a linearisation of the contact model based on dynamically consistent inverse of the contact Jacobians. While the constraint contact model is solved in the slow simulation loop, only a simple model involving six matrix-vector multiplications is applied in the haptic loop.

In order to define the inverse of the generalised contact Jacobians, which is used to evaluate the contact forces in the haptic loop, several assumptions were made. Firstly, by discarding the effect of breaking contact points we assume that the set of active contacts remains unchanged during the update time of the simulation thread. In particular this means that we account for new collisions only at a low update rate. This may lead to delayed perception of the impacts. Similarly, when breaking the contact with the deformable body, a slight perception of attractive forces can appear due to the slow update of broken contacts.

Secondly, to model frictional contact, the set of slipping contacts and the direction of the slip, for each contact, were considered fixed.

Based on our assumptions, we could have considered the matrix $A$ (see Equation 5.36) to be constant during one simulation step. This allowed for the use of the inverted system matrix $\tilde{A}^{-1}$ in the haptic loop, which is necessary for the evaluation of the linearised contact model.

To conclude, we would like to emphasise that this approach defines the contact forces for the haptic thread independently on the number of contact points. Nevertheless, the computation complexity of the contact forces in the simulation loop presents the main bottleneck of the simulation. This includes the construction of the system of linear equations which expresses the velocity constraints and a number of iteration steps needed to solve the FCP problem applying the nested Gauss-Seidel scheme.
5.6 Comparison of the contact force computation models

In this section we will present a comparison of the penalty based and the constraint based models, used for the computation of the contact forces. We will first compare the performance of the two methods in terms of refresh rates, considering the influence of the number of contact points. Then, we will comment on the haptic response obtained during a general interaction between a rigid tool and an elastic object.

In order to investigate the real-time performance of our methods, we performed a task where we pushed a probe at the side of a cylinder and slowly slid around. The cylinder was fixed at the bottom surface and it was deforming under the load applied by the tool. The motion of the probe is depicted on Figure 5.13. We repeated the same task for both contact models, while increasing the number of tracked points defined at the centreline of the tool. The number of initialised points used throughout one run of the task was in a range from three to hundred. To ensure the same input from the haptic interface, the motion of the tool was first recorded with the force feedback turned off. During the test, the acquired sequence was played back in the haptic loop instead of reading the positions and orientations of the haptic device.

The statistical data, obtained from the test, are presented on Figure 5.14. The left graph shows the number of contact points detected throughout the test for each task. On the right graph, relation between the number of initialised tracked points and the running speed of the simulation as well as the haptic loop is presented. For the constraint based method, the haptic refresh rate is maintained at 1 kHz while the update rate of the simulation loop is rapidly decreasing with the increasing

\[ \text{The contact points represent the detected number of contacts. The final number of active contact points which define the force of the constraint based method could be lower.} \]
5. Haptic Rendering

Figure 5.14: Left: Average and maximum number of contact points, detected during the run of each task, in relation to the number of initialised tracked points. Right: The average refresh rates of the simulation and the haptic loop, corresponding to the number of tracked points. All data are shown for both the penalty and the constraint based contact force computation models.

number of contact points. This introduces bigger discontinuities in the contact force values thus diminishing the stability of the interaction. For the penalty based approach, on the contrary, the main bottleneck is the computation in the haptic loop, which slows down the haptic thread with increasing number of contacts. This effect leads to reduced stiffness felt by the user. A reduced speed can be observed in the simulation thread, too. However, we have noticed that this effect is mainly caused by the system scheduler which assigns more computational resources to the haptic loop running with high priority.

To compare the true computational complexity of the two methods, we recorded the net time needed for the contact force generation while performing the aforementioned test. The average computation time corresponding to the average number of detected contact points is shown on Figure 5.15. For the constraint based method this shows the time needed for the detection of the nearest points and for the solving of the FCP using the nested Gauss-Seidel algorithm. These tasks are performed in the simulation loop. For the penalty based method the measured time includes the time spent for the detection of the nearest points (VPs) adjusted to frictional motion and for the computation of the spring-like forces of each contact. In contrast to the constraint method, these tasks are run in the haptic loop. From the graph in Figure 5.15, the better performance of the penetration based method is clearly visible. This can be also justified by considering the implementation of both methods. Due to the use of nested Gauss-Seidel loops, the time needed for one update of the constraint based method depends exponentially on the number of collisions. However, the dependency is only linear for the penalty model. This fact therefore favours the penalty based method for the contact force computation when higher
5.6. Comparison of the contact force computation models

Figure 5.15: Average contact force computation time (in seconds).

amount of contact points is to be used. For small numbers of contact points, both approaches were fast and stable.

Roughly comparing the force output of the two methods, acquired during the test, both models provide plausible force feedback for reasonable contact scenarios (i.e. small penetrations and slow movements without hard impacts). Nevertheless, while the quality of the force response for the constraint based approach is limited mainly by the update rate of the simulation thread, the penalty method exhibits false behaviour stemming from the penetration of the virtual tool into the elastic body. One drawback is the visual aspect of the interaction, since an overlap of the tool and the elastic body is visible (see an example on Figure 5.16). If the penetration depth

Figure 5.16: Comparison of the visual realism when pushing with a tool on a virtual body. Left: The tool and the elastic body. Middle: Interaction using penalty based model (penetration visible). Right: Constraint based model (no overlap apparent).

gets bigger, the tool can even pop through the object, as it was already depicted on Figure 5.10. At this point we have to mention that for the constraint based
method the tool can also sink into the other body. This error is introduced due to the application of velocity constraints for a time-discretised system. Nevertheless, this effect can be visible only when sliding with the tool laterally on the surface for a longer time, while pushing the tool into the object. To avoid this, a small impulse, which eliminates the drift, should be applied in addition to the contact forces.

Finally, the main drawback of the penalty method is the dependency of the contact stiffness on the number of contacts. This issue is sketched on Figure 5.17. The stiffness of the object is perceived smaller when pushing just with the tip of a tool, than when the whole handle of the tool comes into contact with the object. This problem is typical for volume based penalty forces. On the contrary, for the

![Figure 5.17: The dependence of the contact stiffness of the penalty based method on the number of contact points. The contacts appear stiffer when in contact with the whole probe (left), than when only the tool-tip touches the object (right).](image)

constraint based method, always a sufficiently large force for keeping the tool at the surface is determined by the system. The object thus appears equally stiff, independently on the tool interaction style.
Evaluation methods

For applications which haptically model the interactions with virtual deformable bodies, a number of factors exist which influence the quality of the provided haptic feedback. Among them are the selected mechanical deformation model, material laws and the setting of mechanical parameters, collision detection, tool-tissue contact handling, simulation and haptic update rates, coupling between display and simulation, and the characteristics of the haptic device used. Several points in this rendering chain exist, where errors can be introduced, simplifications have to be made, or device limitations are reached. This raises the question of how well haptic sensations, encountered during interaction with real objects, can actually be approximated in a virtual environment.

One approach to determining the fidelity of a rendering method is to perform a Turing-like test, comparing haptic feedback users obtain from real objects against that which they receive from virtual ones which approximate the behaviour of their real counterparts. To this end we designed an experiment, where participants were asked to compare haptic feedback during interaction with real and virtual deformable objects. The virtual model was composed of an enhanced, volume-preserving Mass-Spring system, with parameters tuned according to measurements on real material. Haptic rendering during interaction was performed with a haptic proxy paradigm. We note that the environment was optimised for the specific haptic interface used.

The experimental task was the indentation of a soft object by a rigid tool. We compared the virtual interaction to the real poking of a silicone cylinder with a metal indenter in two tests. While the first one (see Section 6.3) was rather a simple test assessing the fidelity of haptically rendered deformable objects, the second test (see Section 6.4) elaborated on an evaluation technique which would provide better insight into the human perception of the force response during the interaction. More specifically, instead of a simple discriminability measure, which was used in the first experiment, a Multidimensional scaling method was applied for the evaluation of the latter experiment.
This chapter is organised as follows: after presenting the related work on measuring the quality and efficiency of the haptic rendering systems, we will describe our testing environment (see Section 6.2). Then, in Sections 6.3 and 6.4 we will present two experiments which we have performed, i.e. a discrimination and a similarity task. We conclude with the discussion of both experiments and outline the limitations and possible extensions of our studies, which could be addressed in future work (see Section 6.5).

6.1 Related work

Measuring the efficiency, accuracy and realism of haptic virtual environments is a challenging task since it requires consideration of device and user behaviour, evaluation of the force model computation and of the entire force rendering pipeline, as well as an assessment of the perceptual psychophysics. Despite the need for the validation of the entire haptic virtual reality system from a holistic point of view, most previous studies focused on the evaluation of individual components of the haptic rendering system.

One strategy is to judge the suitability of a specific haptic device based upon a number of physical characteristics. In [Hayward and Astley, 1996] several performance measures were defined which are important for a detailed description of the device’s behaviour. Another approach is to assess the efficiency of the haptic systems specific to a given task. Systems used to improve particular motor skills, such as assembly of mechanical machines [Bloomfield et al., 2003], the use of complex tools (i.e. in surgery) [Tendick et al., 2000; H. Esen and Buss, 2003], supporting to facilitate spatial cognition [Feygin et al., 2002], or haptically enhancing virtual environments [Murayama et al., 2004] are evaluated on a base of the time needed for completing the specific task or in terms of a position error measured with respect to a desired path of the interaction. Virtual environments with force-feedback are usually compared against environments lacking force-feedback in order to explore the advantages of the provided haptic cues [Williams et al., 2004]. A combination of quantitative performance measures and subjective user preferences was used to evaluate haptic guidance methods proposed in [Forsyth and MacLean, 2006]. In addition to a performance measure based on mean square error metrics, the suitability of the provided haptic cues was also examined qualitatively based on the answers of users who rated the perceived level of control and the preference of each guidance method. However, rating the realism of the haptic feedback was only a secondary aspect. This is often the case for many studies, where the haptic rendering involves the generation of artificial forces that are not comparable to natural interactions with real-world objects.
To assess the fidelity of a simulation algorithm, as compared to a real-world phenomenon, standard methods from experimental physics can be used. Depending on what the relevant aspects of a simulated interaction are, different data (i.e. positions, forces, time, etc.) recorded during experiments carried out in a real-world setting can be compared with data gathered in virtual simulations. An example of such a comparative assessment was presented by Dupont et al. [2000], who evaluated several friction force computation algorithms and demonstrated better performance of the suggested one. Similarly, in [Ruffaldi et al., 2006], the authors provide “ground truth” data sets consisting of correlated input trajectories and output forces gathered in the real world; these datasets can then be used for an error-metrics-based evaluation of haptic rendering systems. Using root mean squared error metrics on the output force with regard to the real-world measurements, the study assessed the influence of friction compensation, force shading adaptation and mesh resolution on the accuracy of haptic rendering. Although the former evaluation methods guarantee the resemblance of the interaction to the real-world behaviour, they do not address the human perception of the simulated force feedback.

In many cases, high-quality haptic output is not required for ensuring realistic force perception. However, the quality of haptic feedback perceived by the users in virtual environments compared to an interaction with real-world objects has been seldomly investigated. In [Okamura et al., 2003], the realism of virtual biological tissue cutting using scissors was evaluated. Prerecorded forces stemming from physical measurements and forces obtained from linear approximation of the empirical data were compared in a psychophysical study with experienced users. Subjects who were specialised in cutting of the same biological tissues as approximated by the simulations participated in the test. According to the subjects’ responses, none of the generated haptic feedbacks was clearly rated as less reliable. Given the expertise of the participants, the results indicate that both methods closely approximated real feedback.

The evaluation of the overall haptic system as it was perceived by users by direct comparison to a real world reference, has only recently been addressed. One research group [Kuchenbecker et al., 2005] performed a quantitative assessment of the fidelity of an open-loop haptic algorithm for rigid object interaction. Users were asked to tap on given samples holding a PHANToM stylus and rate how realistic the simulation represented the experience of tapping on a real wood sample. In addition to the virtually-driven interaction, real samples of different materials were presented to the participants. The users average realism ratings were evaluated using paired t-tests between all stimuli. Although the hard contact with the wooden sample could not be fully reproduced, the evaluation revealed a significant increase in the virtual surface’s realism when the novel open-loop haptic rendering technique was used.

In our first test (see Section 6.3), we will show a simple method for quantifying the fidelity of a haptic interaction with deformable objects. Two groups of soft objects
will be presented to users: six real silicone samples and six rendered objects. In a discrimination task, users will be asked to determine whether the interaction was virtually driven via the haptic device or whether a real object was present. A signal detection measure, d-prime, was used as an evaluation metric.

However, discriminability may not be the best measure for evaluating haptic rendering fidelity. Another approach to perceptual validation, presented recently in [Cooke et al., 2005, 2006], is to use similarity as a criterion. Similarity has been proposed as a fundamental cognitive process underlying object recognition and categorisation [Edelman, 1999]. Using perceptual similarity between objects as a validation criterion has the advantage of encompassing higher-level, cognitive relationships between objects, which may be more closely correlated with realism and believability than a lower-level measure such as discriminability. Furthermore, MDS techniques can be used to analyse similarity data, yielding rich multidimensional stimulus representations. Thus, the purpose of the second study (see Section 6.4) will be to test whether such an approach could be successfully applied to the problem of haptic fidelity assessment.

6.2 System setup

6.2.1 Computational model

In our experiments we are comparing haptic stimuli, obtained in the real world during an interaction with elastic objects, to those generated by the haptic device in an virtual scenario. We used a Mass-Spring system, as described in Section 4.4.1 as the computation model for the virtual deformable body. The parameters of this model were set in correspondence to the deformation properties of real samples. To obtain fast force response, an explicit numerical integration scheme (see Section 4.4.3) was chosen.

The coupling of the haptic device to the deformation simulation was accomplished using a point-based haptic rendering applying the virtual proxy paradigm as described in Section 5.3. The contact was modelled frictionless.

6.2.2 Reference silicone samples

One of the main drawbacks of the MSS is the process of setting the deformation parameters. Spring constants, masses, and mesh topology have to be adapted to obtain a specific deformation behaviour. In order to achieve realistic behaviour with our model, we determined the deformation parameters from reference silicone samples. To obtain a material with nearly linear-elastic properties, we used a two-part silicone rubber called ECOFLEX (Smooth-On, www.Smooth-On.com). We
applied different mixtures of the Ecoflex 0030, Ecoflex 0040 and silicone thinner to prepare several cylindrical rubber phantoms for our studies. The approximate dimensions were $\varnothing 80 \text{ mm} \times 80 \text{ mm}$, with a slight difference in height (up to 3 mm) across the samples, due to a manual manufacturing.

An aspiration test [Nava et al., 2003] was performed in order to determine the material properties of the silicone objects. According to these tests, the silicone phantoms could be considered as a neo-Hookean material with Young’s moduli in a range of 16 to 78 kPa, with the Poisson ratio assumed to be 0.499.

According to the measured properties of the real samples, virtual objects were set up. In Figure 6.1 a uniformly tetrahedralised model of our virtual object is shown. This defined the mesh of the MSS, consisting of 300 nodes, 1656 edges and 1156 tetrahedra. We used the methods described by Gelder [1998], Deussen et al. [1995] and Delingette [1998] to set the stiffness, mass and the time step parameters of the MSS, respectively (see Sections 4.4.2, 4.4.2 and 4.4.3 for details). Node and spring damping parameters were manually selected to ensure that the dynamic deformation remained stable.

![Figure 6.1](image)

**Figure 6.1:** The tetrahedral mesh of the virtual cylinder, consisting of 300 nodes, 1656 edges and 1156 tetrahedra.

Finally, the constants for the virtual coupling to the proxy object were selected. This connection was set as stiff as possible while still allowing stable interaction. The values varied from $0.7 \text{ N/mm}$ to $1.5 \text{ N/mm}$. 
6.2.3 Hardware setup

We chose SensAble’s PHANToM device to provide the interface for our virtual environment. This decision was made on the basis of availability. Due to the limitations in the device’s force response [Cavusoglu et al., 2002; Campion and Hayward, 2005], the PHANToM was not the optimal device for our task. One significant problem we experienced was the limited maximum forces of the device. From empirical tests, this was estimated to be around $4.2 \, N$ for a long-term peak force. Therefore, choosing the PHANToM device presented possible drawbacks for a faithful display of the complete stiffness range of our virtual objects. Nevertheless, with some adaptations of hardware and software (as will be described later on), we were able to provide adequate feedback in all cases. We wish to emphasise, that our analysis did not focus on evaluating any specific device, but rather the fidelity of the complete haptic interaction environment.

The test application was run on a Linux PC with 2xP4 2.8GHz processors. We used two asynchronous threads: one for haptic force rendering using the proxy model and one for deformation calculation. During the experiments the haptic thread was running at 1kHz, while the deformation calculation was running with an approximate refresh rate of 100Hz.

6.3 Discrimination task

In order to evaluate the quality of haptic rendering we carried out a discrimination experiment related to indentation of real and virtual objects. We set up a scenario, where the subjects had to discriminate between the categories of real and virtual objects only by pushing on the presented samples with a rigid tool. A discriminability measure was applied to express the differences quantitatively.

6.3.1 Reference samples and virtual objects

The discrimination study was processed with six cylindrical rubber phantoms, manufactured as described in Section 6.2.2, with elasticity varying between 20 $kPa$ and 35 $kPa$. Between all real samples there were slight differences in their stiffness and their height. The stiffness variation was 35%, while the maximum difference in the sample height was 3 $mm$.

Additionally, six virtual cylinders were set up for the test. At first, one virtual object was set to match the deformation properties of one specific silicone sample, taking into account the parameter settings described in Sections 4.4.2 and 4.4.2. In the following these objects are referred to as R2 and V3, for the real and the
virtual sample respectively. Then, five variations of the tuned virtual model V3 were created, to resemble the differences in the real objects. They were adjusted to cover the range of heights and stiffnesses of the real samples. An overview of the considered parameters of all objects is given in Table 6.1. Note that, due to the addition of volume preserving forces, we reduced the elasticity modulus of the virtual samples. These adjustments were made in order compensate for the stiffness of the volume preserving forces.

<table>
<thead>
<tr>
<th></th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
</tr>
</thead>
<tbody>
<tr>
<td>elasticity [kPa]</td>
<td>20.6</td>
<td>23.5</td>
<td>27</td>
<td>28.6</td>
<td>29</td>
<td>35.6</td>
</tr>
<tr>
<td>stiffness ratio</td>
<td>1.14</td>
<td>1</td>
<td>0.87</td>
<td>0.82</td>
<td>0.81</td>
<td>0.66</td>
</tr>
<tr>
<td>height [mm]</td>
<td>80</td>
<td>81.5</td>
<td>82.5</td>
<td>81</td>
<td>79.5</td>
<td>80.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Virtual stimuli</th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
</tr>
</thead>
<tbody>
<tr>
<td>elasticity [kPa]</td>
<td>14.5</td>
<td>17</td>
<td>17</td>
<td>20.4</td>
<td>21.2</td>
<td>25.5</td>
</tr>
<tr>
<td>stiffness ratio</td>
<td>1.17</td>
<td>1</td>
<td>1</td>
<td>0.83</td>
<td>0.80</td>
<td>0.66</td>
</tr>
<tr>
<td>height [mm]</td>
<td>82</td>
<td>80</td>
<td>82</td>
<td>80</td>
<td>81.5</td>
<td>78.5</td>
</tr>
</tbody>
</table>

**Table 6.1**: Estimated elasticity (Young’s modulus in kPa) and heights (in mm) of the samples used during the experiment. The indicated ratio is the elasticity ratio computed in reference to the object R2 and V3 for the real and virtual samples, respectively.

The different force responses of the real and virtual samples were measured and are presented in Figure 6.2. These measurements were made only approximately using the PHANToM device. All forces were determined from the position, where the gravity of the indenter was counterbalanced by the deformation force of the current sample, and thus at zero force response little deflection can be noticed. These measurements are not very precise and thus just the differences of the slopes of each curve should be compared.

### 6.3.2 Interaction overview

Participants were asked to push with a metal ball indenter onto a real silicone cylinder, or on the virtual object respectively. The whole task was performed blindly. After several indentations, users were asked whether they believed to have touched a real or virtual object. Because free tool-object interaction would involve many contact effects (e.g. friction, multiple contact points, torques, etc.), which would influence the interaction, the whole procedure was limited to a one-dimensional manipulation. Touching the body was possible just in one point (the middle of the object’s top surface), and only in vertical direction. The forces rendered on the haptic device have also been restricted to this direction. However, contact forces
Figure 6.2: The recorded forces of the real (a) and virtual (b) samples. The drift on the height axis for the real, as well as the virtual samples indicates the variation (around 3 mm) in the heights of the samples.

6.3.3 Experimental apparatus

The experiment was performed with a PHANToM haptic device as previously described. A 250 mm long pen-shaped stylus was attached at one end to the robot arm of the PHANToM device. On the other end of the pen a ball intender was attached. In order to find a trade-off between substantial penetration depths and pure elastic deformation of the real samples, we used a ball indenter with a radius of 4 mm. This allowed indentation depths of up to 20 mm while still being able to reproduce the same behaviour with the PHANToM device. The stylus was sliding inside a 30 mm long tube, equipped with ball bearings to ensure resistance free motion. This limited the interaction to one direction. To avoid parasitic vibrations coming from the rigid side-grip of the stylus inside the tube, the computed PHANToM forces have been projected to the same direction. Beneath the indenter, six real samples were placed on a rotatable support while leaving place for virtual samples. A blocking system of the rotatable plate allowed for fast and precise positioning of the middle of each sample below the indenter. The whole setup was put behind a tall barrier to prevent the user from observing the apparatus during the test. A complete view of the test setup can be seen in Figure 6.3.
Figure 6.3: Experimental setup with a PHANToM, attached indenter driven in a tube, and a rotatable sample support.

Participants sat at a table and put their hand through an opening in the barrier. Through the opening the users were holding the pen-shaped indenter. During the whole test the participants could rest their arm on a support. In the setup the user could freely move the indenter up and down for 40 mm, without getting into contact with the surrounding construction. The indentation depth was limited to approximately 20 mm by a rigid stop, due to the mentioned force rendering limitations of the PHANToM device. To mask the sound of the PHANToM motors and the ball bearing, users wore closed headphones as acoustic ear protectors with white noise played. No additional visual feedback, e.g. from the virtual simulation, was provided to the user, thus each participant had to make her decision just according to the haptic stimuli she received. The operator was on the other side of the barrier monitoring the progress and changing the samples beneath the indenter.

6.3.4 Experimental procedure

The experiment consisted of two phases: training session, and testing. The objective of the training phase was two-fold. Firstly, a user should get used to the setup and the haptic feedback in general. This included the limits of the desired interaction style. Participants were instructed to slowly approach the objects with
the indenter, always hold the indenter in their hand during the interaction, perform just slow movements (approximately two punches per second) and move within the non-restricted range of 20 mm indentation depth (before hitting the rigid stop).

Secondly, the two categories of real and virtual objects were presented, so that the user can determine the differences between both classes. To this end, he could experience the interaction between one real sample and a virtual one, namely the objects R2 and V3 whose elastic properties have been tuned to match. This phase lasted till the participants felt comfortable with the apparatus and were ready for the main test.

During the testing phase, subjects were told that they will be presented several real objects with small differences in stiffness and height. Additionally, virtual objects, with similar deformation characteristics to the real objects, will also be shown among the real samples. The user was informed, that the task is to express an estimate if interacting with a real or a virtual object.

For each subject the test consisted of 50 trials, where a randomly selected object from the six real and six virtual models was presented to the user within each trial. No response feedback was given to the participants after each trial.

### 6.3.5 Participants

Thirteen participants (two female, eleven male), took part in the test. Their age ranged from 26 to 54 years (with an average of 30 years). All but two subjects were right-handed (although just one was using the left hand for the tests). None of them reported any known haptic deficit due to an accident or illness. Two participants were experienced PHANToM users. While most of the others users already experienced haptic simulations with the PHANToM before, they can not be considered as experienced users of haptic devices.

### 6.3.6 Results

All participants went through the learning phase within five minutes, and completed the experiment of 50 trials on average in 20 minutes. Four different combinations of trial condition and response were possible. Hits occurred, when a real sample was correctly identified, and false alarms, when a virtual object was assumed to be real. Similarly, a correct rejection was counted, when a virtual object was recognised as such, and a miss was recorded, when a real object was attributed as virtual.

Counting false alarms and misses, the mean number of wrong responses of all participants was 18.66 in 50 trials, with a standard deviation of 7.15. To further analyse the data (see for instance [Stanislaw and Todorov, 1999]), we start with determining
hit rate $H$ and false alarm rate $F$. The former describes the probability of a correct detection, when a real object was present, while the latter denotes the probability of assuming a real object, when a virtual sample was shown. In the optimal case, hit rate is high and false alarm rate low, while values around 50% indicate decision making by chance. Figure 6.4 a) depicts the results of our study. The mean values of hit and false alarm rate are 62.8% and 36.3%, respectively.

Two further measures have to be determined to analyse the data - sensitivity (also referred to as discriminability) and response bias. The former denotes the ability to detect the actual category of real or virtual objects, and the latter a possible tendency towards reporting a specific category more often.

As described in [Stanislaw and Todorov, 1999], we determine the non-parametric measure of sensitivity $A'$ according to

$$A' = \begin{cases} 
0.5 + \frac{(H - F)(1.0 + H - F)}{4.0H(1.0 - F)} & \text{if } H \geq F \\
0.5 + \frac{(F - H)(1.0 + F - H)}{4.0F(1.0 - H)} & \text{if } H < F
\end{cases}$$

The measure usually ranges from 0.5, which indicates chance performance, to 1.0, which corresponds to perfect category detection. Values less than 0.5 may arise due to response confusion. The non-parametric measure of bias $B_D''$ can be determined by

$$B_D'' = \frac{(1.0 - H)(1.0 - F) - HF}{(1.0 - H)(1.0 - F) + HF}.$$  

Positive values represent a tendency to report interaction with the virtual object category, while negative values represent a tendency to report the real one. Non-existent bias is indicated by values close to 0.0. Both measures are shown in Figure 6.4 b). Mean sensitivity is 0.707 with a mean bias of 0.01.

While we are still above chance level, the results show, that subjects were unable to perfectly discriminate between the two categories. This is also reflected by comments gathered from the participants, who regarded correct discrimination of the categories as rather difficult.

To further analyse our results, we investigated whether a specific sample could be easier recognised than the others. However, no indication of this could be noticed. The mean of wrong recognition of real samples was 37.3%, with a standard deviation of 5.15%; and the one of virtual samples 36.1%, with a standard deviation of 4.74%.

We also examined, if a change in performance could be noticed during the course of the experiment. Only in two cases, the ability to differentiate between virtual and real objects improved slightly in the second half of the test. Nevertheless, this has not been statistically significant.
Apart from the quantitative data, after the trials we also collected comments from participants to determine their approach during the test. We found that the best detection performance was achieved by participants who reported a small difference during the first contact of the probe with the sample. It was described as a “sinking of the tool” effect. We attribute this phenomenon to the selected proxy-point simplification of the haptic rendering which generates forces linearly dependent on the penetration depth. Nevertheless, for spherical indenter the contact force depends nonlinearly on the indentation depth, i.e. \( F = c \delta^3 \) (for a depth \( \delta \) and a material specific stiffness \( c \), see also [Mahvash et al., 2002]). Clearly, for the sphere model the contact stiffness is increasing gradually whereas for the point-based model a bigger discontinuity in the contact force stiffness is apparent. The use of the simple model can therefore cause the differences felt by the users when touching the surface only slightly. However, all users reported the discrimination task to be very hard for deeper penetrations.

Other cues, which were sometimes present, were small vibrations encountered during interaction with the virtual model. This was due to the alignment of the indenter with the ball bearing. A small rippling effect could be noticed, which became more evident with the tilting of the moving indenter. Rendered forces could lead to such a tilt, since the direction was not perfectly aligned with the driving tube.

Moreover, two subjects reported that a low frequency wave was noticeable for a moment after stopping the indenter inside the sample. Also, this behaviour was more prominent for harder objects. This is probably due to the limited information propagation speed of the applied explicit integration scheme. An indication of this effect can also be observed in an example force profile recorded during an interaction with a virtual object as shown in Figure 6.5. However, in general, the recordings did not reveal clear differences between the two models apparent in positions, velocities, or smoothness of forces.

**Figure 6.4.** Statistical plots of all participants: a) Hit versus False alarm rate, b) Bias versus Sensitivity. The mean values are marked with a square.
6.3. Discrimination task

Finally, none of our participants actually reported the haptic rendering during interaction as unrealistic or artificial.

6.3.7 Conclusions

In this study we examined the fidelity of a simple deformation model for providing haptic feedback during interaction with virtual elastic objects. Deformation parameters were determined based on reference silicone samples. A discrimination task was carried out, in which participants had to differentiate between real and virtual objects. Results showed that this task was quite complex, since only small differences between real and virtual haptic feedback could be noticed. Thus, we were able to achieve a high fidelity of virtual rendering.

We can also infer, that our approach for parameter tuning of our deformable models is at least sufficient to provide reasonably realistic haptic feedback. Nevertheless, an extrapolation to complex objects may not be straightforward.

This experiment revealed also several shortcomings of our system. For the hardware setup, the mechanism used for guiding the indenter introduced high-frequency noise. On one hand this degraded the realism of the virtual simulation, on the other hand it could also mask the behaviour when interacting with the silicone sample thus making the two environments less distinguishable. A better solution should be therefore found for the guiding mechanism.

Furthermore, limitations of our haptic rendering approach were observed. Due to the simplifications of the contact model, which did not account for the spherical
shape of the indenter, differences of the force feedback during the initial phase of tool-tissue contact were detected.

### 6.4 Similarity-based validation experiment

Our second study elaborated on the shortcomings of the first experiment. At first, the hardware setup was improved by changing the tool guiding mechanism and by increasing the maximum output force. Also, we used a flat cylindrical indenter for which the contact force depends linearly on the indentation depth similarly as it is for the point-based haptic rendering. Nevertheless, the main contribution was in using a more rigorous evaluation method for the comparison of perceptual differences in haptic rendering techniques.

We applied the **multidimensional scaling** (MDS) technique for the analysis of similarity ratings provided by users comparing pairs of haptically-presented objects. Similar as in our first study, but unbeknownst to the participants, real and virtual deformable objects were presented to the users. In addition, virtual objects were either presented under higher-fidelity rendering condition or under lower-fidelity condition in which force filtering and proxy-point filtering were removed. We hypothesized that reducing fidelity of virtual rendering would exaggerate the difference between real and virtual objects.

Within this study we demonstrated how MDS analysis can provide an opportunity to visualise and quantify the perceptual effects of changes in rendering parameters and how it can be used in the evaluation of haptic rendering scenarios.

#### 6.4.1 Multidimensional scaling technique (MDS)

MDS techniques are a family of algorithms which take pair-wise proximities between objects (e.g., distances between pairs of cities) as input and return the coordinates of the objects embedded in a multidimensional space (e.g., a geographical map showing the cities’ relative positions). MDS has been used extensively by psychologists in the study of a wide range of multivariate stimuli, allowing for the identification of important psychological dimensions of stimulus variation and quantification of perceptual distances between stimuli, e.g., Shepard [1963]; Melara and Day [1992]; Hollins et al. [1993]; Hsu et al. [2005]. Using human similarity ratings on a set of objects as the input proximity data, the output configuration can then be interpreted as a map of the objects in a psychological space which explains the similarity data Borg and Groenen [2005]. MDS provides information about the **dimensionality** of the psychological representation (i.e., how many stimulus dimensions play a role in human similarity ratings and what the dimensions’ relative contributions are), as
well as more specific topological information about the representations, such as the relative spacing between objects along a given dimension.

We used MDS to assess the fidelity of haptic interactions. As proximity data, we used similarity ratings provided by users who haptically probed a set of real and virtual objects. Using MDS, we wished to test whether real and virtual objects varying in stiffness would be separable along one or more perceptual dimensions. For a truly high-fidelity interaction, real and virtual objects should be indistinguishable and, if the objects vary solely in terms of stiffness, MDS should recover a single perceptual dimension corresponding to perceived stiffness. For lower-fidelity interactions, more than one perceptual dimension may emerge from the interaction, e.g., a dimension along which real and virtual stimuli are separated.

### 6.4.2 Haptic rendering conditions

The MDS was used to compare two virtual haptic environments, which we shall refer to as higher-fidelity and lower-fidelity conditions. The two environments differed in terms of force and proxy-point position filtering (for details see Section 5.3). In the higher-fidelity environment, we linearly blended the position of the proxy point within five steps. Recall that this interpolation was required due to a difference in refresh rates: in our system the haptic loop runs at 1 kHz, while the simulation loop runs at 100 Hz. Although the proxy point was always locally adjusted within the haptic loop, large differences in its position appeared at each simulation step when the surface triangles of the deformed body moved. Interpolating between proxy point positions helped to mitigate this effect. Moreover, for enhanced stability we also filtered the last five values of the computed force vectors using a discretised Gaussian filter before the display on the haptic device.

In the lower-fidelity environment, we turned off the proxy point position blending. Furthermore, we simply averaged the last five samples of the force vector before rendering the force on the device. While the underlying deformation algorithm stayed the same, these changes led to a reduced fidelity of the haptic rendering. This manifested itself in minute force discontinuities and slightly diminished overall stability. Although this effect is difficult to describe, one can think of it as a reduction of the overall smoothness of the interaction. The resulting difference between the two conditions was not obviously perceptible and could only be readily detected by experienced users of haptic interfaces.

### 6.4.3 Reference samples and virtual objects

Within our similarity test we used seven silicone samples, manufactured as described in Section 6.2.2, with Young’s moduli of 16, 20, 29, 38, 46, 69, 78 kPa. The maximum difference in sample height was 2.5 mm.
In addition to the real samples, we created seven virtual objects for our experiments. Each of the seven virtual samples was tuned according to the material properties of one of the seven manufactured silicone cylinders. The same deformation parameters were used for virtual models in both the lower and higher-fidelity rendering conditions. This study did not require that the stiffness of real and virtual objects be exactly matched. However, the virtual samples were designed to roughly cover the same stiffness range as the real samples as quantified by their Young’s moduli. It should be noted, that the overall force response of the virtual object is a result of several factors, including the mechanical model, as well as the virtual coupling to the haptic device. For comparison, Figure 6.6 shows the force response of the virtual samples, which include all such factors, together with the force response of the real objects. Note that due to differences in the heights of the silicone samples, the force output measured on real samples starts at different height for each sample. The corresponding stiffness ratios of the stimuli, as measured from the plots in Figure 6.6, and the height differences are listed in Table 6.2.

Figure 6.6: Force response of real samples (left), measured during an indentation experiment with 8 mm flat cylindric indenter, and simulated force output of the virtual samples (right). The drift on the height axis for the real samples indicates a slight variation (at most 2.5 mm) in the heights of the samples. The height of the virtual samples was calibrated to the height of the hardest real sample (stimulus 7). The corresponding stiffnesses of the plots and the height differences of the real samples are given in Table 6.2.

6.4.4 Hardware limitations

The main limitation of the chosen PHANToM device, is the maximum long-term peak force, estimated to 4.2 N. When this value is exceeded, force feedback is switched off and the application driving the PHANToM stops. To keep the application running, we limited the computed force to the maximum force which can
be applied by the haptic device. The maximum value of 4.2 \( N \) guarantees realistic force response for indentations up to 6 \( mm \) for the hardest sample. For penetrations deeper than 6 \( mm \), the force output was held constant. To increase the interaction depth (i.e., by increasing the maximum force output of the haptic device), an additional weight of 230 \( g \) was attached on the other end of the PHANToM arm, directly on one of its motors. This weight constantly pulled the PHANToM stylus upward, which provided an additional increase of the maximum force by about 2.3 \( N \) in the upward direction. For our experiment, we could therefore render forces up to 6.5 \( N \) which was sufficient for indentations in the hardest sample of up to 9 \( mm \). Table 6.3 shows the maximum allowable indentation depths for each virtual object.

<table>
<thead>
<tr>
<th>Virtual stimuli</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Young’s modulus [kPa]</td>
<td>16</td>
<td>20</td>
<td>29</td>
<td>38</td>
<td>46</td>
<td>69</td>
<td>78</td>
</tr>
<tr>
<td>Maximum depth [mm]</td>
<td>43</td>
<td>32</td>
<td>22</td>
<td>19</td>
<td>17</td>
<td>11</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 6.3: The Young’s modulus (in kPa) parameter of the virtual samples (stimuli 8 – 14) and maximum depth of realistic interaction (in mm). Above these values, rendered forces would exceed the maximum force limit of the haptic device, and would be therefore held constant at 6.5\( N \).

Attaching an additional weight to the PHANToM arm required a real-time, software-driven gravity compensation. For both interactions, with real and virtual objects, a force which counter-balanced the additional weight was generated on the haptic device throughout the whole experiment. When a virtual object was being presented, this force was added to the forces stemming from the computation model of the virtual sample. The gravity compensation force was played back from a pre-recorded force profile. Forces, sufficient to maintain the position of the PHANToM stylus, were recorded at 23 positions spread vertically across the interaction axis and stored as the gravity compensation force profile.
6.4.5 Experimental stimuli: real and virtual objects

Participants performed similarity ratings amongst a set of fourteen objects. This set consisted of seven silicone cylinders and seven virtual cylinders, which varied in stiffness. The same set of real and virtual objects was used for the whole experiment. However, half of the subjects experienced the virtual objects under the higher-fidelity filtering condition, while the other half experienced them under the lower-fidelity one.

Importantly, participants were not told about the differences amongst the objects being presented (real/virtual, soft/hard). The real samples were not visible to the participants during the experiment; they were hidden under a carton box as shown in Figure 6.7 (right). On each trial, the rod slid through a small hole in the box, presenting the same visual feedback to the participant. The PHANToM device, attached to the indenter, was not hidden from the users. Participants, who asked about the hardware were told that it was used to record indenter positions, velocities, and applied forces during the interaction.

The experiment was restricted to the study of one-dimensional interactions with soft objects. This choice was made to minimise the influence of contact effects like friction, multiple contact points, torques, etc. Prodding of the soft bodies was only possible on the top surface of the cylinder, around the middle of the top surface, and only in vertical direction.

Since the deformation model was optimised for quasi-static interaction, participants were instructed to limit their indentations to slow movements. This resulted in contact velocities of about 100 mm/s on average. The rationale for doing so was to avoid high velocity contacts, which would cause complex dynamic effects that might not be accurately reproducible with the mechanical model.

6.4.6 Experimental apparatus

The experiment was performed with a PHANToM haptic device, with a 250 mm long pen-shaped rod attached at one end to the PHANToM robot arm (Figure 6.7). A flat cylindric intender of 4 mm radius was attached on the other end of the pen. The flat indenter was chosen to alleviate the need of shape compensation within the force response algorithm. The choice of the size of the cylinder was selected due to a trade-off between substantial penetration depth and elastic deformation of the real samples. We instructed the participants to keep the maximum interaction depth around 20 – 25 mm. A mark was put on the stylus pen which was to stay always above the keyhole of the construction. However, no rigid stop was preventing the users from pushing deeper into the object. Participants were asked to get used to
the interaction depth during the training session, however, during the experiment itself they were encouraged not to watch the mark.

The stylus was held upright by three cylinders. They ensured smooth motion of the indenter, limited to one (vertical) direction. To avoid sideways vibrations due to the stiff contact of the stylus and the cylinders, the computed PHANToM forces have been projected along the vertical direction. Beneath the indenter, seven real samples were placed on a rotatable support while leaving room for virtual samples. All the samples, and the tip of the indenter, were covered by a carton box, upon which the participant could lean his/her arm while performing the experimental task (see Figure 6.7, right).

![Figure 6.7: Hardware setup exposed (left) and covered for the experiment (right). It consisted of a PHANToM, an indenter attached to the PHANToM, three cylinders to prevent sideways motion, and a rotatable plate for the silicone samples.](image)

To mask the sound of the PHANToM motors, users were asked to wear closed headphones as acoustic ear protectors. No visual feedback, e.g. from the virtual simulation, was provided. The operator was on the other side of the table monitoring the progress, and changing the samples beneath the indenter. The experimenter’s motions were kept as consistent as possible across trials and were the same regardless of whether the target object was real or virtual.

### 6.4.7 Experimental procedure

The experiment consisted of two phases: a training session and a testing session. In both phases, the participants’ task was the same: they performed indentations on two objects, one after another, and then rated the similarity between the two objects on a scale between one (low similarity) and seven (high similarity) by typing the number on the keyboard. In the training phase, participants first practised using
the setup and interacting with the objects until they were able to prod the objects as directed. Participants held the metal rod between their thumb and their index finger of the right hand. They were instructed to slowly approach the objects with the indenter, to hold the indenter in their hand during the interaction, and to try to keep the maximum indentation depth at around 20 mm. All of the objects were introduced to the participants so that they could indent them, notice the differences, and establish their rating scale. Participants were told that similarity judgements should not take the slight difference in sample height (max. 2.5 mm) into account. In the training phase, every participant went through the same sequence of presented stimuli. The training phase lasted for about 20 minutes, during which it was possible to perform at least 50 trials. At the end of the training session, all participants felt comfortable with the task and the setup, and were ready to run the main test.

The testing phase consisted of five blocks of similarity ratings. In each of the five blocks, all pairwise combinations of the fourteen objects (105 combinations) were presented to the participant in random order. On each trial, the experimenter set the first object below the indenter by turning a plate with all samples to the right position (Section 6.4.6) or by altering appropriate settings for the virtual objects, and then asked the user to test the sample. The user was allowed to prod the sample just once. After the user lifted the indenter, the experimenter presented the second object, and again asked the user to prod the sample and provide the similarity rating. One testing phase lasted about 3 h. During the whole test, the movement of the indenter (positions and velocities), as well as the forces generated by the virtual model were recorded at 100 Hz for later analysis.

After completing the experiment, participants were asked to fill in a questionnaire in which they were requested to explain how they performed the similarity ratings, and to describe the samples used in the test.

6.4.8 Participants

Ten right-handed participants (19 – 43 years old) were paid 32 EUR to perform the experiment. None of them reported any haptic deficit due to accident or illness. None of the participants had used a kinaesthetic haptic device before and all of the users were naive about our research in the haptic field.

6.4.9 Multidimensional scaling analysis

Subjects’ similarity ratings (values ranging from 1 to 7) were converted to dissimilarities (values ranging from 0 to 1) and the average pairwise dissimilarities were computed over all subjects for each rendering condition. The mean dissimilarity data were analysed separately for each condition using the non-metric MDS algorithm.
6.4. Similarity-based validation experiment

(MDSCALE) implemented in MATLAB. As opposed to the classic metric MDS techniques, which fit exact proximity values, non-metric MDS techniques only take the rank-order of the pair-wise proximity values into account and are thus better suited for human similarity data. The algorithm searches for an output configuration in a space of fixed dimensionality and returns a goodness-of-fit measure, Kruskal’s S1 Stress, which is the normalised difference between the fitted distances in the output configuration and the observed proximities Cox and Cox [2001]. Stress values are used to determine the required dimensionality of the output configuration. Given our set size of 14 objects, stress values below 0.2 can be taken as an indication that the dimensionality of the output space is sufficient to faithfully represent the input proximities Borg and Groenen [2005].

Applying MDS analysis on similarity data in this setting provides a quantitative method for identifying perceptual differences between real and virtual haptic objects, as well as between objects rendered using the two different rendering methods. The object sets we used included both real and virtual samples which were expected to vary in stiffness. Thus, for each of the rendering methods, we used MDS to test whether subjects’ similarity ratings could be explained by a unidimensional configuration, where the single dimension corresponded to stiffness, or whether additional perceptual dimensions (e.g., a systematic difference between real and virtual stimuli) are necessary to explain the similarity data. In addition to the question of overall dimensionality, we were interested in visualising output configurations in order to better understand the perceptual differences between individual samples.

6.4.10 Results

Similarity data from each of the participants was grouped according to the rendering condition under which they experienced the haptic objects and the average similarity was taken over all participants in each group (Figure 6.8). Although patterns of changing similarity can be seen in these data (e.g., stimulus 1 is highly similar to stimulus 8), performing MDS analysis of the similarity data provides a much richer perspective, which includes the following kinds of information:

1. how many dimensions of variation in the stimuli were apparent to the participants;

2. whether one of these dimensions corresponded to the material property being simulated (stiffness, in this case);

3. whether one or more undesirable perceptual dimensions were also apparent to the subjects (e.g., a dimension along which real and virtual stimuli are separated);
4. the relative weights of the desired dimensions relative to undesired perceptual dimensions;

5. interstimulus distances in the perceptual space.

Figure 6.8: Mean similarity ratings for subjects in the higher-fidelity rendering condition (left) and the lower-fidelity condition (right). The mean standard error taken over all pairwise proximities was 0.59 in the higher-fidelity condition and 0.55 in the lower-fidelity condition.

Perceptual dimensions of MDS configurations

The first question addressed by MDS analysis is that of the appropriate dimensionality of the output configuration. This is classically determined by inspection of the MDS stress plot, as described in Section 6.4.9. Figure 6.9 shows the stress plots for MDS solutions with dimensionalities of 1 – 5 obtained using mean similarity from the higher-fidelity and the lower-fidelity condition. In the former, stress drops below the 0.2 threshold for a one-dimensional solution, indicating that one perceptual dimension is sufficient to explain the similarity data. In contrast to this, for the lower-fidelity condition, stress was 0.25 for a one-dimensional solution, and drops sharply to 0.05 for a two-dimensional one. This sharp decrease, which is more evident for the lower-fidelity condition, clearly indicates that two perceptual dimensions are required to explain the similarity data.

MDS does not provide labels for the dimensions of the output configuration; these must be interpreted by visual inspection. The output configurations generated by two-dimensional MDS solutions are shown in Figure 6.10 for the higher-fidelity (left) and the lower-fidelity condition (right). Note that for ease of comparison, the configurations shown here have been scaled and rotated using the Procrustes transform implemented in MATLAB such that the points representing the real objects were
Figure 6.9: Stress as a function of the dimensionality of the MDS solution computed using similarity data from each of the two rendering conditions. A dashed line is drawn at a stress level of 0.2; configurations with stress below 0.2 are considered to adequately explain similarity data.

brought as close as possible to equidistant points along the x-axis. For both conditions, the first perceptual dimension recovered by MDS (plotted along the x-axis) clearly corresponded to stiffness.

As mentioned above, in the higher-fidelity condition, the single dimension of stiffness sufficed to explain the similarity data. This finding provides support for the perceptual validity of the higher-fidelity environment, insofar as differences between real and virtual objects did not play a significant role in determining perceptual similarity.

Figure 6.10: Two-dimensional MDS maps for the higher fidelity condition (left) and lower fidelity condition (right).
In contrast to this, a second perceptual dimension is required to explain similarity data in the lower-fidelity condition. From the lower-fidelity configuration we see that this dimension causes a perceptual separation between the real and virtual samples, and that the separation increases with stiffness. Note that a similar, albeit much smaller trend is also visible in the higher-fidelity condition. This trend also increases at higher stiffness levels. Comparing the results from the two conditions, it appears that the lower-fidelity condition exaggerated or compounded a difference between real and virtual stimuli which was also present in the higher-fidelity condition, but only had a negligible perceptual effect.

**Topology of MDS configurations**

The MDS maps also provide more detailed topological information about stimulus ordering and clustering in perceptual space. First, participants were able to perfectly recover the rank order of real samples along the stiffness dimension in both rendering conditions. In addition, the spacing between real samples is similar under both conditions, indicating that subjects’ perception of the real samples’ stiffness was not influenced by the rendering mode of virtual samples. Participants also recovered the rank order of virtual stimuli to a large extent, although stimuli 10 and 11 were perceived as almost equally stiff in the high-fidelity condition, while stimulus 14 was perceived as slightly less stiff than stimulus 13 in the lower-fidelity condition. However, differences in stiffness levels were not equal between all samples. For the real samples, there was a greater perceptual difference in stiffness between samples 2 and 3 and between samples 5 and 6 than between other neighbouring pairs. These differences were constant across rendering conditions. Virtual stimuli cluster in roughly the same way, yielding a group of lower stiffness objects (real: 1, 2, virtual: 8, 9), a group of moderate stiffness objects (real: 3 – 5, virtual: 10 – 12), and a group of higher stiffness objects (real: 6, 7, virtual: 13, 14).

Interestingly, this clustering pattern is in accordance to the one predicted by the Young’s modulus of the samples, i.e., the spacing between the samples in the map matches the differences between the object stiffnesses quite well. However, the absolute range of perceived stiffness is compressed for the virtual objects compared to the real ones. In addition, the degree of compression is different in the two rendering conditions: the stiffest virtual objects (13, 14) have a perceived stiffness equivalent to that of real object 4 in the lower-fidelity condition, while their stiffness is closer to that of real object 5 in the higher-fidelity condition.

A plot of recorded positions and computed force responses is shown in Figure 6.11. It shows a typical interaction with three representative virtual samples of the low, moderate, and high stiffness groups.
6.4. Similarity-based validation experiment

Figure 6.11: Positions and force response recorded during the interaction with three representatives of virtual objects: low stiffness stimulus 8 (top left), moderate stiffness stimulus 11 (top right) and high stiffness stimulus 14 (bottom). The computed force is indicated with dotted line; it was cut off at 6.5 N due to the PHANToM’s maximum force limitations. The rendered force, perceived by the users, is drawn with dashed line; it overlaps with the computed force for the softer objects, but becomes smaller than the computed force for the stiffer objects. The recordings were captured at 100 Hz.

Debriefing questionnaires

In the questionnaires, participants were asked to describe the objects and to explain how they performed similarity ratings. Interestingly, none of the participants detected the presence of real and virtual objects in the study. All but one assumed that all samples were real objects; the other participant thought that all samples were virtual. His assumption was likely due to the fact that he was aware of the existence of haptic device technology and assumed that all stimuli would be presented using the device.

In their descriptions of the objects, participants mentioned several properties which they had used for similarity judgements, most of which could be related to stiffness. For example, participants described objects as being soft and homogeneous and
compared them to foam, rubber, jelly, sponge, or metal springs. The fact that stiffness-related terms were mentioned so often confirms that the first dimension in the MDS maps can be interpreted as stiffness.

A smaller number of participants described objects as having a rigid skin or shell, with a softer interior. This was sometimes attributed as hollowness. One interpretation of this could be that it was just a different description of the stiffness dimension. However, it is also possible that this relates to a second, unexpected perceptual dimension in the experiment. The hollowness attribute might have been caused by the maximum force limitation of the haptic device. When large penetration depths were used during interaction, the displayed force had to be held constant, which led to a sudden pop into the sample and which may have induced a sense of “hollowness”. In order to rule out the possibility that this could correspond to the second dimension we found in the MDS plots, we grouped the participants according to the penetration depths they used during interactions and performed another MDS analysis of these data. However, the results showed that penetration depth did not cause the separation of objects along the second dimension.

Nevertheless, participants did not mention differences in haptic rendering fidelity in the questionnaires. These effects were of course quite subtle and could not easily be detected, but it could have been expected that for the lower-fidelity condition users would report another criteria which they used for their similarity assessment. Clearly, further work is needed in this direction, especially requiring a larger number of participants.

6.4.11 Conclusions

The MDS analysis revealed that participants were able to recover stiffness changes across both real and virtual objects, in both lower-fidelity and higher-fidelity haptic rendering environments. We found a noticeable perceptual difference between real and virtual objects in the lower-fidelity condition, especially at high stiffness levels. In contrast to this, in the higher-fidelity environment real and virtual objects were quite similar especially at low stiffness levels.

Although real and virtual objects showed a trend towards differentiation along a second dimension with increasing stiffness, this dimension was found to be perceptually more significant in the lower-fidelity condition. Plotting the objects in a two-dimensional perceptual space also revealed that stiffness differences roughly correlated with differences in Young’s moduli. Finally, the verbal reports allowed us to confirm that stiffness was, indeed, the most important perceptual dimension in both rendering conditions.

The MDS analysis revealed that two perceptual dimensions should be used to explain the data in the lower-fidelity condition. Along the second dimension the virtual
Discussion and outlook

In this chapter we presented two studies on quantifying the fidelity of haptically rendered deformable bodies. In both cases, the evaluation was based on comparison of the virtually generated haptic stimuli to those obtained from real world.

In our pilot study, concentrating on the discrimination task, we found that subjects could discern between the categories of real and virtual stimuli with a mean accuracy of 63% with no significant bias towards assuming the presence of either real or virtual objects. While being above chance level, the results indicate that we were able to approximate haptic feedback of a real object with high fidelity in our specific hardware setup.

Our second study elaborated on the evaluation method of the discrimination task. We have shown how Multidimensional scaling analysis of human similarity judgements on a set of real and virtual objects can be used to gain both quantitative and qualitative insight into the relationships between haptic rendering parameters and human perception. In this specific test, two haptic rendering conditions were compared. MDS analysis revealed a clear perceptual distinction between real and virtual objects in the lower-fidelity condition, but not when relying on higher-fidelity rendering. In addition, MDS maps of the objects in human perceptual space showed that real and virtual objects are most similar at lower levels of stiffness and become more distinguishable as stiffness increases. Removing the smoothing of proxy-point positions and rendered forces compounded this effect.

One of the issues we wanted to address in our studies was the evaluation of a virtually generated haptic interaction with soft bodies, which could be applied for training of prospective surgeons. While the selected tissue model was too simple to be used in surgical simulation, the experiment already indicates that a reasonable approximation of real behaviour can be reached. In this respect, one also has to consider that the participants in our study were explicitly told to look for small differences. During surgical simulation, slight deviations from perfect feedback might be acceptable, since the trainee does not fully focus on small discrepancies. Since the main target of surgical simulation should be to achieve a training effect, if
and to what degree a small deviation from real feedback would affect this process still remains an open question. However, a final answer to this problem is beyond the scope of this thesis. It should also be emphasised that we do not suggest to use such simple deformation models for a surgical training system. The study only determined, how well forces coming from a real object can actually be approximated. Tests with more complex deformation models should also be carried out.
Applications

In this chapter we will present three applications where our haptic rendering approaches were implemented. The first one describes a simple testing environment. The other two demonstrate the integration of our force feedback algorithms in research projects focusing on surgical simulation and augmented reality.

7.1 Testing setup

This environment was designed for testing our haptic rendering approaches. The primary interest was in simulation of frictional contacts between a soft body and rigid tools and in generation of stable and realistic force feedback. The deformable virtual objects were also visually displayed to the user, however, visual appearance was only of secondary interest.

The main virtual scenario consists of an elastic body, modelled using a Mass-Spring system as described in Section 4.4.1, rigid tools and rigid walls. Due to the limitations of our collision detection system (see Section 4.4.4), the deformable body must be of a simple, in general convex shape. Nevertheless, our algorithm can also cope with nonconvexities, as long as the tool moves close to the surface. The elastic object can be either fixed at several nodes, or it can move freely in space, subject to gravity and contact forces. Collision detection handles contacts of the soft body with the tool or the walls. Currently, haptic feedback can only be determined for a single tool. Frictional contact is included in all collisions. To compute the interaction forces for the tool contact we use the penalty or the constraint based methods, described in Chapter 5. The contact between the elastic body and the rigid walls is modelled using the velocity based constraint technique (see Section 5.5).

In Figures 7.1, 7.2 and 7.3 three main interaction scenarios are depicted. Figure 7.1 shows an example of deforming a block of soft tissue fixed at one side, by a probe haptically modelled with points set on the centreline. This setup represents the
7. Applications

**Figure 7.1:** Deforming a block of tissue: pushing at the edge with the handle of a probe and pulling at the corner of the elastic block.

**Figure 7.2:** Peg-in-hole scenario with elastic base and rigid tool. For the haptic response the tool is modelled by 68 contact points.

**Figure 7.3:** Tool interaction with frictional contact. Flipping the cylinder, by pushing on it from the side, and rolling the cylinder by sliding the tool over the top.
main scenario in which the developed haptic algorithms were tested. All our haptic rendering approaches (with and without applied friction) provide smooth and stable force feedback when touching the flat areas of the deformable object. Nevertheless, at the sharp edges discontinuities in the force output can be felt. These are caused by point sampling of the line segments defining the virtual tool. In addition, for the penalty based method it is difficult to correctly handle the interaction at sharp edges. Since a constant distance between the tracked surface points is not maintained, the tool can easily be pushed into the object. Eventually the tool can even pass through the whole body with only little resistant force felt (see Figure 5.10). For the constraint based method this problem does not apply.

To demonstrate the interaction with arbitrary shaped tools sampled with multiple points at the surface, a “peg-in-hole” scenario with a mushroom-like tool has been set up (see in Figure 7.2). Although the deformable body in this scenario is not convex, the collisions could be detected correctly. This is because of the circular shape of the hole, for which the nearest surface points can be still correctly tracked even if the tool is moved inside the hole. With a big amount of points which define the virtual tool (68 points), where most of them can be simultaneously in contact with the deformable body, this setup serves as a good benchmark for the efficiency of the applied force computation methods. Moreover, the modelling of tight contacts, as is the case when the peg is inserted into the hole, represents a computationally difficult task, on which the stability of the provided haptic feedback can be tested. As a matter of fact, some instabilities in the force feedback have been observed for the constraint based method, when applying frictional contact forces for the tool-tissue interaction. The main cause of these instabilities is the slow update of the inverse contact Jacobians, which is due to the low efficiency of the algorithm solving the nonlinear, frictional contact problem. The speed of this algorithm depends exponentially on the number of detected contacts. Applying the penalty based method, which complexity is linear with respect to the number of contacts, the interaction has been stable.

Finally, the last example, depicted in Figure 7.3, shows a scene for which the necessity of including frictional forces into the contact force computation is demonstrated. It shows an interaction between a probe tool and a loose cylinder, where the goal was to flip the cylinder and roll it on the ground. If sufficient friction is applied between the cylinder and the ground, it is possible to flip the cylinder by pushing it at the side, close to the top surface. Pushing close to the bottom surface would make the cylinder slip away. At last, applying friction forces between the tool and the cylinder allows for rolling the cylinder only by sliding the tool parallel to the ground over the object. This interaction would not be possible without the modelling of frictional forces.
7.2 Hysteroscopy simulator

Virtual reality based surgical training is one of the main application areas where simulation of contacts between rigid tools and soft tissues is necessary. Therefore, we have chosen the simulation of surgical interventions as one of the demonstration scenarios for our haptic rendering approaches. As a base environment for the implementation, a hysteroscopy simulator [Harders, 2006], developed in our institute, was used. In this training system the surgeon can perform minimally invasive hysteroscopic interventions like removing of tumours (e.g. polyps, myomas) from inside the uterus. During the training, contacts between the tool and the uterine cavity as well as the tool and the tumours are simulated.

In the simulator the surgical instruments are approximated by a collection of points. To detect the first collisions between the tool points and the cavity with embedded polyps and myomas, a spatial hashing collision detection algorithm [Teschner et al., 2003] is used. This avoids the limitations of our previous method, i.e. tracking the nearest features, applied for collision detection. The detected collision points are tracked according to the tracing algorithm (see Section 4.4.4), while they are in contact. This step is necessary since the spatial hashing algorithm has been designed to detect collisions with the entities (i.e. triangles or tetrahedra) which were crossed by the tool points within the last time step. Thus, once the tool is immersed in the organ, it is not possible to detect the surface points where the virtual tool entered the tissue. The tracking of the last contact points, detected at the surface of the deformable body, is therefore necessary. This is provided by our tracking algorithm. Once the surface points are obtained, the contact forces are generated using the penalty based model for multiple contact points, described in Section 5.4.

In Figure 7.4, the shape of a cutting loop, as modelled in the hysteroscopy simulator and its approximation by six collision points, used for the generation of the haptic feedback, is shown.

The hysteroscopy training system was designed to run the simulation and the haptic device on separate machines, connected via intranet. The contact forces between the model and the surgical tool are computed on the simulation site. The limited update rate of about 250 Hz on the current hardware (4x AMD Opteron 2.2GHz) presents a problem for the haptic rendering. Instabilities of the force feedback can emerge for higher stiffnesses. To mitigate this effect, a local model is used on the haptic client side. For each simulation step, an average contact plane, defined by the set of current contact points, is sent to the client. Forces are then obtained based on the distance of the tool tip to the plane. Moreover, to avoid discontinuities between consecutive transmitted planes, a smooth transition is achieved by blending plane positions and normal vectors in the haptic loop.

Applying the penalty based method in this setting presents difficulties for the interaction with the polyp. If the tool is modelled only by a few points, set sparsely at
the cutting loop, the user can pop through the polyp with the surgical instrument if higher force is applied. Using the constraint based technique would overcome this problem. However, for the current implementation this would introduce a considerable computational overhead. Nevertheless, we believe that the efficiency of this method can be substantially boosted by parallelising the iterative Gauss-Seidel method. The inclusion of the constraint based approach is thus planned for the near future. Note that, although the penalty based method is applied at the moment, still a realistic interaction can be provided (i.e. avoiding the pop-through effect) when the tool is reasonably densely sampled with points and when the stiffness of the penalty force is set high enough.

### 7.3 Augmented reality

As another demonstration environment for our haptic rendering system, it was integrated in an augmented reality prototype. Using the setup developed by Bianchi et al. [2006], we created a scenario where the user could observe and interact with real and virtual cylinders set in the same scene. For realistic haptic as well as visual feedback, the silicone cylinders and their tuned virtual counterparts presented in the evaluation experiments (Chapter 6) were used. The single contact point limitation of our evaluation studies applied also to this setup. The user could therefore push on the body only with the tip of a pen-shaped tool. In Figure 7.5 (left), the view

**Figure 7.4:** Example scene of a polyp and a cutting tool (left). Detected collision points of a simplified tool set out of six points and corresponding contact forces applied (right).
of the real scene with an augmented image displayed on the screen is shown. The interaction with the virtual cylinder is shown in Figure 7.5 (right).

Figure 7.5: The augmented reality application. The real scene with the user’s view shown on the screen (left) and close-up views of deforming the virtual object (right).

The main problem, which emerges in this setup is the visible penetration of the real tool in the virtual sample. This is specific to the selected method of computing the contact forces. This issue is not that obvious in pure virtual environments, like the presented hysteroscopy simulation, since the generated scene does not have to be collocated with the real world. Nevertheless, for applications in open surgery, where the surgeon also sees his hands during the intervention, this represents a major limitation. To avoid this problem, different ways of coupling the haptic device to the virtual simulation could be investigated. Nevertheless, due to the hardware limitations, i.e. slow update rate and low Z-width of the haptic device, a different strategy would be required. The most reasonable solution is to include a separate deformation model for the visual output, which would enforce boundary conditions at the contact points according to tool displacement.
Conclusions and outlook

8.1 Contact force computation

In this thesis we investigated the computation of contact forces between rigid tools and elastic bodies, for the purpose of displaying them to the user via haptic devices. The main application area of this research is virtual reality based surgical training systems, where the contacts between surgical instruments and organs are simulated. A main target was thus the design of effective haptic rendering algorithms for tool-tissue interaction.

Despite the large number of existing surgical simulators, the contact models used have been kept simple. Simplifications considering only single point contacts or soft tissue interactions with line segments are commonly applied. Nevertheless, a number of projects exist which model the interaction with tools of complex geometries. These, however, are limited by the applied contact force model which does not consider friction. Although recently methods for haptic rendering of frictional contact forces between deformable bodies were presented, this is still an active research field. Mainly optimisation techniques which enable fast update rates and methods which enhance the transparency of haptic feedback are sought for.

With respect to this current research, we investigated methods for simulation of frictional contacts for multiple collision points.

8.1.1 Haptic rendering with multiple contact points

We proposed two techniques for the haptic rendering of tool-tissue contacts: a penalty based method, which extends the point based god-object paradigm to multiple, independently tracked points; and a constraint based approach which models the god-object as a rigid body of complex geometry. In both cases, frictional contact was considered.
The first method promises a fast contact force generation, where most of the computation can be performed in the haptic loop. This increases the stability of the haptic rendering. Nevertheless, this approach suffers under the presence of a permanent penetration of the virtual tool into the soft tissue. Under high load this issue can lead to unrealistic behaviour like pooping through the virtual object. Virtual coupling of the haptic device to the virtual tool was applied to lessen this effect. However, the presence of some penetration is required by the contact force generation algorithm and thus cannot be completely avoided.

To handle the non-penetration contact constraint correctly, a different approach was investigated. We improved the contact force model presented by Otaduy and Gross [2007], which solves for contact forces by satisfying velocity-based contact constraints defined for a set of points. We adapted the proposed approach to include non-equality constraints and frictional effects, similarly to the solution described by Duriez et al. [2006]. In contrast to Duriez et al. [2006], our approach applies velocity based Signorini contact conditions, which present advantages over the position based conditions when simulating collisions with rigid tools. Nevertheless, the main contribution is in the adaptation of the contact force linearisation process to compensate for static as well as dynamic friction. This process defines the parameters necessary for fast update of the force feedback, run in the haptic thread.

During the comparison of the two suggested methods (see Section 5.6), several examples were presented which favour the constraint based method over the penalty based one, if considering the physical correctness of the generated contact forces. The most important features are that the virtual tool does not penetrate the deformable object while in contact and that the contact stiffness is independent of the number of detected contact points, which does not hold for the penalty based method. Moreover, the setting of the necessary parameters for the constraint based approach is also easier, since it does not include artificial properties like the penetration force stiffness. Only hardware and simulation specific values, like the coupling stiffness and the simulation and haptic loop update rates, are necessary.

Another advantage of the constraint based method is the multirate design of the haptic rendering algorithm. Only few data are required to compute the contact force in the haptic loop. The amount of the necessary data is constant and does not depend on the number of tracked contact points, which is the case for the penalty based approach. Therefore, this model should be preferred for systems, which run the haptic device and the simulation system on separate machines and where the communication between these two systems is to be kept minimal.

The main disadvantage of the constraint based method is that the passivity can be compromised if the parameters used in the linearised contact model are updated slowly. These are set in the simulation loop according to the force response of the soft body. This makes the stability of the provided haptic feedback dependent on the
complexity of the deformable body. In contrast to this, the computational complexity of the penalty method depends solely on the number of tracked contact points. Therefore, for scenarios involving complex deformation modelling, the penalty based approach promises better stability of the haptic feedback.

At last we wish to note that both described methods present realistic approaches for a real-time 6-DOF haptic rendering of tool-tissue contacts with friction.

8.1.2 Future work

For both methods the number of contact points, which can be handled in real-time, presents the main bottleneck. To accelerate the contact force computation, parallelisation of the algorithms could be applied. This would be particularly effective for the penalty based method, where the adaptation of the nearest surface points can be processed independently for each contact point. For the constraint based model, the computation of the nonlinear complementarity problem, with its exponential dependence on the number of contact points, is the least effective part of the algorithm. Therefore, different solvers should be tested for faster convergence.

In the design of the contact model, collisions were considered only for a predefined group of points, set at the rigid tool. This simplification induces non-smooth force output, when sliding with the tool over a sharp edge. Adapting our algorithms to account also for collisions between edges of the virtual objects would overcome this problem.

8.2 Realism measurement

Haptic rendering of deformable objects is a critical element of several multimodal virtual reality applications which are used for instance in the medical, entertainment, and education sectors. The main trade-off in selecting appropriate rendering approaches for deformable objects is that of real-time capability vs. fidelity. This often leads to the key question of how realistic a computer-based approach can be in simulating a real object.

To this end, we constructed an experiment, where the users could compare the haptic feedback received from an interaction with real objects against that obtained from a computer generated virtual model, resembling the real scenario. The presented pilot experiments were limited to an indentation task, where the users were prodding the presented samples with a metal rod.

Two evaluation techniques were applied for the validation of haptic rendering algorithms. For both the human perception of real objects was used as a benchmark.
The results of our experiments indicate that a reasonable approximation of real
behaviour was reached in our specific settings. Indirectly comparing two render-
ing algorithms it was also shown that a subtle reduction of the smoothness of the
force output enhances the perceptual differences between the real and the virtual
interaction.

8.2.1 Future work

In the future we would like to extend our evaluation studies to investigate a wider
range of rendering parameters and interaction styles (i.e. allowing for fast and oscil-
latory movements, and strong impacts in order to investigate the dynamic properties
of the simulated bodies). The next level of experiments should allow for less con-
strained tool-object interaction, thus considering the influence of the tool shape
and frictional characteristics. We want to point out that rather than an evaluation
of the frictional parameters settings, further tests could provide valuable input on
the importance of the frictional effects, e.g. the stick-slip or creep behaviour (see
Section 5.3.3), on human perception of the friction forces.

In our studies, only a simple mass-spring model has been used for the deformation
modelling of the elastic bodies. Although in this research we investigated haptic
rendering methods which could be applied in surgical simulation, we do not suggest
to use such simple deformation models for surgical training systems. Therefore, tests
with more complex methods should also be carried out.

Finally, we would like to point out the importance of verbal responses of the partici-
pants. These can provide explanations of differences detected during the evaluation
studies. Together, these studies would provide valuable insight into haptic perception
of virtual deformable objects.
For the purpose of calculating the stiffness parameters for our MSS model, discussed in Section 4.4.2, and for formulating the implicit integration rules presented in Section 4.4.3 we have to linearise the force functions for a given configuration (i.e. node positions). Therefore, the Jacobian of the force function, also known as the stiffness matrix of the system, is to be formulated:

\[ K_F := J_{p,F} = \frac{\partial F}{\partial \mathbf{p}}. \]

The stiffness matrix also represents a negated Hessian of the system, as was referred to in Section 4.4.3:

\[ H_{p,ij} = \frac{\partial F_i}{\partial \mathbf{p}_j} = -\frac{\partial^2 E_C}{\partial \mathbf{p}_i \partial \mathbf{p}_j}. \]

According to our definition of the elastic forces, based on energy function formalism \( E_C = \frac{1}{2} k \sum C(\mathbf{p})^2 \), for a given condition \( C(\mathbf{p}) \) (see [Baraff and Witkin, 1998]), the stiffness matrix is expressed by:

\[ K_{FE,ij} = \frac{\partial F_i}{\partial \mathbf{p}_j} = -k_{FE} \left( \frac{\partial C(\mathbf{p})}{\partial \mathbf{p}_i} \frac{\partial C(\mathbf{p})}{\partial \mathbf{p}_j}^T + C(\mathbf{p}) \frac{\partial^2 C(\mathbf{p})}{\partial \mathbf{p}_i \partial \mathbf{p}_j} \right). \]  

\[ (A.1) \]

The above equation has to be summed across all force elements \( FE \) in the system (i.e. springs or tetrahedra), in order to obtain the stiffness matrix element \( K_{ij} \) of the entire system:

\[ K_F = [K_{ij}] = \sum_{FE} K_{FE,ij}. \]

Note, that each matrix \( K_{ij} \) is a \( 3 \times 3 \) matrix, as all calculations are done in 3D Euclidean space.

### Stiffness matrix of the linear elastic spring forces

We recall our definition of the elastic spring force condition (see Equation 4.4):

\[ C_S(\mathbf{p}) = C_S(\text{EDGE}_{ij}) = \| \mathbf{p}_i - \mathbf{p}_j \| - l_0. \]
Using Equation A.1 we can express the elements of the corresponding stiffness matrix for the linear spring \( S = \{i, j\} \):

\[
K_S = \begin{pmatrix}
\frac{\partial F_S^i}{\partial p_i} & \frac{\partial F_S^i}{\partial p_j} \\
\frac{\partial F_S^j}{\partial p_i} & \frac{\partial F_S^j}{\partial p_j}
\end{pmatrix} = \begin{pmatrix}
-A_S & A_S \\
A_S & -A_S
\end{pmatrix},
\]

\[
A_S = \frac{\partial F_S}{\partial p_j} = -k_S \left( -\frac{(p_i - p_j)}{\|p_i - p_j\|} (p_i - p_j)^T \right) + (\|p_i - p_j\| - l_0) \frac{\partial^2 C(p)}{\partial p_i \partial p_j}, \quad (A.2)
\]

where the second mixed derivative is given by:

\[
\frac{\partial^2 C(p)}{\partial p_i \partial p_j} = \frac{1}{\|p_i - p_j\|} I + \frac{1}{\|p_i - p_j\|} \frac{(p_i - p_j)}{\|p_i - p_j\|} \frac{(p_i - p_j)^T}{\|p_i - p_j\|}. \quad (A.3)
\]

Note, that for the stiffness parameter identification (see Section 4.4.2), the term expressed in Equation A.3 can be omitted, since the analysis is done around the equilibrium state. For the equilibrium state of the material, the condition \( C(p) \) equals zero. Nevertheless, for the numerical integration all terms have to be used since the system is linearised at each time step. The Equation A.2 can be simplified to:

\[
A_S = -k_S \left( -I + \frac{l_0}{\|p_i - p_j\|} I - \frac{l_0}{\|p_i - p_j\|} (p_i - p_j) (p_i - p_j)^T \right). 
\]

**Stiffness matrix of the volume preserving forces**

The condition penalising for the volume change was defined as (see Equation 4.7):

\[
C_V(p) = C_V(TET_{ijkl}) = V - V_0 = \frac{1}{6} (p_i - p_j) \cdot ((p_i - p_j) \times (p_k - p_j)) - V_0.
\]

The stiffness matrix of the volume preserving forces \( F_V \) is a 12 × 12 matrix:

\[
K_V = \begin{pmatrix}
\frac{\partial F_V^i}{\partial p_i} & \frac{\partial F_V^i}{\partial p_j} & \frac{\partial F_V^i}{\partial p_k} & \frac{\partial F_V^i}{\partial p_l} \\
\frac{\partial F_V^j}{\partial p_i} & \frac{\partial F_V^j}{\partial p_j} & \frac{\partial F_V^j}{\partial p_k} & \frac{\partial F_V^j}{\partial p_l} \\
\frac{\partial F_V^k}{\partial p_i} & \frac{\partial F_V^k}{\partial p_j} & \frac{\partial F_V^k}{\partial p_k} & \frac{\partial F_V^k}{\partial p_l} \\
\frac{\partial F_V^l}{\partial p_i} & \frac{\partial F_V^l}{\partial p_j} & \frac{\partial F_V^l}{\partial p_k} & \frac{\partial F_V^l}{\partial p_l}
\end{pmatrix}.
\]

Applying Equation A.1, the element \( K_V,ij \) of the corresponding stiffness matrix for the volume preserving force is:

\[
K_{V,ij} = -\frac{k_V}{6} \left( \frac{1}{6} \mathbf{A}_i \mathbf{A}_j^T + (i \neq j) (V - V_0) \mathbf{p}_i \mathbf{p}_j^* \right),
\]
where $\mathbf{A}_i$ denotes a vector perpendicular to the face opposed to the node $i$ and is
given by the outer product $((\mathbf{p}_l - \mathbf{p}_j) \times (\mathbf{p}_k - \mathbf{p}_j)) = ((\mathbf{p}_l - \mathbf{p}_k) \times (\mathbf{p}_j - \mathbf{p}_k))$. The
orientation of the vector $\mathbf{A}_i$ is such that $\mathbf{A}_i \cdot (\mathbf{p}_i - \mathbf{p}_j) > 0$. The term $(\mathbf{p}_l - \mathbf{p}_k)^*$
represents a cross product in a matrix-vector form:

$$
\mathbf{p}^* = \begin{pmatrix}
0 & -p_z & p_y \\
p_z & 0 & -p_x \\
-p_y & p_x & 0
\end{pmatrix}
$$

In addition, the term $(\mathbf{p}_k - \mathbf{p}_l)^*$ does not contribute to the diagonal elements $K_{V,ii}$, which is expressed by the condition $(i \neq j)$.

For the convenience, we give now the full expression for the stiffness matrix $K_V$
of a tetrahedra TET$_{ijkl}$ with vertices oriented according to the right-hand rule
(i.e. $\text{Volume}(\text{TET}_{ijkl}) = \frac{1}{6}(\mathbf{p}_i - \mathbf{p}_j) \cdot ((\mathbf{p}_l - \mathbf{p}_j) \times (\mathbf{p}_k - \mathbf{p}_j)) > 0$):

$$
K_V = -\frac{k_V}{36} \begin{pmatrix}
\mathbf{p}_{jl} \times \mathbf{p}_{jk} & \mathbf{p}_{jl} \times \mathbf{p}_{jk} \\
\mathbf{p}_{ki} \times \mathbf{p}_{ki} & \mathbf{p}_{ki} \times \mathbf{p}_{ki} \\
\mathbf{p}_{lj} \times \mathbf{p}_{li} & \mathbf{p}_{lj} \times \mathbf{p}_{li} \\
\mathbf{p}_{ij} \times \mathbf{p}_{ik} & \mathbf{p}_{ij} \times \mathbf{p}_{ik}
\end{pmatrix}
$$

$$
- \frac{k_V}{6} (V - V_0) \begin{pmatrix}
0 & \mathbf{p}_{ik}^* & \mathbf{p}_{jk}^* \\
\mathbf{p}_{ki}^* & 0 & \mathbf{p}_{ik}^* \\
\mathbf{p}_{lj}^* & \mathbf{p}_{ij}^* & 0
\end{pmatrix}
$$

where we used the notation $\mathbf{p}_{ij} = (\mathbf{p}_j - \mathbf{p}_i)$. 
In order to process the numerical integration of the MSS, as described in Section 4.4.3, we need to compute the Jacobians \( J_{p,ij} \) and \( J_{v,ij} \) of the internal force functions. The corresponding relation of the Jacobian \( J_{p,ij} \), for the elastic part of the inner forces, i.e. \( \frac{\partial (F_S + F_V)}{\partial p} \), was given in Appendix A. The Jacobian \( J_{v,ij} \) for the elastic part of the inner forces is zero, as it does not depend on the velocities of the mass nodes.

In the following we will therefore derive the Jacobian matrices of the damping forces, \( J_{p,ij} = \frac{\partial F_D}{\partial p} \) and \( J_{v,ij} = \frac{\partial F_D}{\partial v} \), which can be used for the implicit and semi-implicit numerical integration presented in Section 4.4.3.

We recall our definition of the spring damping force between nodes \( i \) and \( j \) (see Equation 4.3):

\[
F_D = -k_D \dot{C}(p) \frac{\partial C(p)}{\partial p},
\]

which is given for the condition:

\[
C(p) = \|p_i - p_j\| - l_0.
\]

The time derivative of the above condition is therefore:

\[
\dot{C}(p) = \frac{(p_i - p_j)}{\|p_i - p_j\|} \cdot (v_i - v_j).
\]

The partial derivatives of the damping function can be expressed as (see [Baraff and Witkin, 1998]):

\[
J_{p,ij} = \frac{\partial F_i}{\partial p_j} = -k_D \left( \frac{\partial C(p)}{\partial p_i} \frac{\partial \dot{C}(p)^T}{\partial p_j} + \dot{C}(p) \frac{\partial^2 C(p)}{\partial p_i \partial p_j} \right),
\]

\[
J_{v,ij} = \frac{\partial F_i}{\partial v_j} = -k_D \frac{\partial C(p)}{\partial p_i} \frac{\partial \dot{C}(p)^T}{\partial v_j} = -k_D \frac{\partial C(p)}{\partial p_i} \frac{\partial C(p)^T}{\partial p_j}.
\]
Using Equations A.2 and A.3 we can express the Jacobians $J_{p,D}$ and $J_{v,D}$ of the damping force, defined for the spring $S = \{i, j\}$:

$$J_{p,D} = \begin{pmatrix} \frac{\partial F_p}{\partial p_i} & \frac{\partial F_p}{\partial p_j} \\ \frac{\partial F_p}{\partial p_i} & \frac{\partial F_p}{\partial p_j} \end{pmatrix} = \begin{pmatrix} -A_{p,D} & A_{p,D} \\ A_{p,D} & -A_{p,D} \end{pmatrix},$$

$$J_{v,D} = \begin{pmatrix} \frac{\partial F_v}{\partial v_i} & \frac{\partial F_v}{\partial v_j} \\ \frac{\partial F_v}{\partial v_i} & \frac{\partial F_v}{\partial v_j} \end{pmatrix} = \begin{pmatrix} -A_{v,D} & A_{v,D} \\ A_{v,D} & -A_{v,D} \end{pmatrix},$$

where the second mixed derivative is given by:

$$\frac{\partial^2 C(p)}{\partial p_i \partial p_j} = \frac{1}{\| p_i - p_j \|} I + \frac{1}{\| p_i - p_j \|} \frac{(p_i - p_j) (p_i - p_j)^T}{\| p_i - p_j \| \| p_i - p_j \| \| p_i - p_j \|}.$$

$$A_{p,D} = -k_D \left( -\frac{(p_i - p_j)}{\| p_i - p_j \|} (v_i - v_j)^T + \left( \frac{(p_i - p_j)}{\| p_i - p_j \|} \cdot (v_i - v_j) \right) I \right) \frac{\partial^2 C(p)}{\partial p_i \partial p_j},$$

$$A_{v,D} = k_D \frac{(p_i - p_j)}{\| p_i - p_j \|} \frac{(p_i - p_j)^T}{\| p_i - p_j \| \| p_i - p_j \|},$$

where the second mixed derivative is given by:
Solving a 3D frictional contact problem

The computation of constraint based contact forces (see Section 5.5) requires to solve a nonlinear complementarity problem stated in Equation 5.36. We will now describe two iterative fixed point methods, namely the projective and the block Gauss-Seidel method, which will be applied to obtain a solution for a single contact point and multiple contact points respectively. Our approach follows the approach presented in [Renouf and Acary, 2006]. However, for a better understanding of the principles of the applied iterative solvers one is referred to [Cottle et al., 1992].

C.1 Single contact point

Considering one contact point we need to find \( \tilde{\lambda} = [\tilde{\lambda}, \tilde{\zeta}] = [\lambda, \zeta^1, \zeta^2] \) being a solution of the frictional contact problem \( FCP(A, b) \):

\[
\begin{align*}
FCP(A, b) : \quad & w = A\tilde{\lambda} - b \geq 0, \\
& w_n \geq 0, \quad \tilde{\lambda} \geq 0, \quad \mu_s\tilde{\lambda} - \|\tilde{\zeta}\| \geq 0 \\
& w_n \perp \tilde{\lambda}, \quad w_n \perp \|\tilde{\zeta}\| , \quad \|w_\tau\| \perp \|\tilde{\zeta} - \mu_D\tilde{\lambda}\_i\|w_\tau\|, \quad (C.1)
\end{align*}
\]

where the matrix \( A \) describes the relationship between the normal \((n)\) and the tangential \((\tau = [t_1, t_2])\) forces for a single contact point:

\[
A = 
\begin{pmatrix}
A_{n,n} & A_{n,t_1} & A_{n,t_2} \\
A_{t_1,n} & A_{t_1,t_1} & A_{t_1,t_2} \\
A_{t_2,n} & A_{t_2,t_1} & A_{t_2,t_2}
\end{pmatrix}.
\]
We solve for $\tilde{\lambda}$ by projected Gauss-Seidel method. Let $A = B + U$, with $B$ being lower triangular and $U$ strictly upper triangular parts of matrix $A$. At each iteration we will solve the constraint problem:

$$FCP(B, b^{\nu}) : B\tilde{\lambda}^{\nu+1} - b^{\nu} \geq 0$$

for

$$b^{\nu} = b - U\tilde{\lambda}^{\nu},$$

given an approximate solution $\tilde{\lambda}^{\nu}$ and keeping the constraints of the original FCP. Since $B$ is a lower triangular matrix, the problem $FCP(B, b^{\nu})$ can be solved explicitly by a forward substitution. The new solution $\tilde{\lambda}^{\nu+1}$ has to satisfy the inequality conditions of Equation C.1, which will be obtained by a projection of $\tilde{\lambda}^{\nu+1}$ onto a feasible region:

$$\tilde{\lambda}_i^{\nu+1} = \text{proj} \left( A_{i,i}^{-1}(b_i^{\nu} - \sum_{j=1}^{j-i-1} B_{i,j}\tilde{\lambda}_j^{\nu+1}) \right),$$

$$b_i^{\nu} = b_i - \sum_{j=i+1}^{j=3} U_{i,j}\tilde{\lambda}_j^{\nu}.$$

The projection function $\text{proj}(\cdot)$ is defined as:

$$\text{proj}(\tilde{\lambda}) = \begin{cases} \tilde{\lambda}, & \text{if } \tilde{\lambda} \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

$$\text{proj}(\tilde{\zeta}) = \begin{cases} \tilde{\zeta}, & \text{if } \|\tilde{\zeta}\| \leq \mu S\tilde{\lambda} \\ \mu D\tilde{\lambda} \frac{\tilde{\zeta}}{\|\tilde{\zeta}\|}, & \text{otherwise} \end{cases}.$$

This iterative procedure is repeated, until convergence. Currently we check for the absolute error between $\tilde{\lambda}^{\nu}$ and $\tilde{\lambda}^{\nu+1}$.

### C.2 Multiple contact points

To solve for the contact problem $FCP(A, b)$ with $m$ contacts, we will apply a block splitting scheme and solve for the collision impulses contact by contact. In the block splitting method, the matrix $A$ is split into submatrices $A_{\alpha,\beta}$ of the order $3 \times 3$. Each of the submatrices carries the relationship between contact points $\alpha$ and $\beta$, for the normal and tangential space. The submatrices on the diagonal thus describe FCP for a single point, for which the solver described in previous section can be used.
The idea of solving the frictional problem with multiple contact points is to solve iteratively for each contact a local FCP, $FCP(A_{\alpha}, b_{\alpha})$, while considering the contribution of the other contacts to be fixed. Block splitting of the matrix $A$, $A = B + U$, for the lower block triangular matrix $B$ and the strictly upper block triangular matrix $U$, is applied:

$$B = \begin{pmatrix} A_{1,1} & 0 & 0 & \cdots \\ A_{2,1} & A_{2,2} & 0 & \cdots \\ \vdots & \vdots & \ddots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,m} \end{pmatrix},$$

$$U = \begin{pmatrix} 0 & A_{1,2} & A_{1,3} & \cdots \\ 0 & 0 & A_{2,3} & \cdots \\ \vdots & \vdots & \ddots \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$  

Subsequently, this block scheme is solved by an iterative block Gauss-Seidel approach. In every loop of the block Gauss-Seidel method we compute the collision free velocities $b_{\alpha}$ for each contact point and solve the local contact problem with friction. The local FCP is formulated as:

$$FCP(A_{\alpha}, b_{\alpha}^\nu): w_{\alpha} = A_{\alpha,\alpha} \tilde{\lambda}_{\alpha}^{\nu+1} - b_{\alpha}^\nu \geq 0,$$

for the collision free relative velocities approximated by fixing the other contacts:

$$b_{\alpha}^\nu = b_{\alpha} - \sum_{\beta < \alpha} B_{\alpha,\beta} \tilde{\lambda}_{\beta}^{\nu+1} - \sum_{\beta > \alpha} U_{\alpha,\beta} \tilde{\lambda}_{\beta}^\nu. \quad (C.2)$$

A schematic description of the algorithm computing the contact forces for multiple contact points with friction reads:

**Step 1:** Process the iterative block Gauss-Seidel method to obtain a solution $\tilde{\lambda}^{\nu+1}$, starting from $\tilde{\lambda}^0 = \tilde{0}$:

**Step 1.1:** for each contact $\alpha = 1, \ldots, m$ do

**Step 1.2:** Compute the free velocity update, $b_{\alpha}^\nu$, for contact $\alpha$ following Equation C.2

**Step 1.3:** Solve local frictional problem $FCP(A_{\alpha}, b_{\alpha}^\nu)$ by applying the iterative projected Gauss-Seidel scheme as described in Section C.1.

**Step 2** Check for convergence of the global solution $\tilde{\lambda}^{\nu+1}$, and return to **Step 1** if necessary
Bibliography


Alberty, J., Carstensen, C., Funken, S. A. and Klose, R. [2002], ‘Matlab implementation of the finite element method in elasticity’.


Dahl, P. R. [1968], A solid friction model, Technical Report ADA041920, AEROSPACE CORP EL SEGUNDO CA.


Debunne, G., Desbrun, M., Cani, M.-P. and Barr, A. [2001], Dynamic real-time deformations using space and time adaptive sampling, in ‘Computer Graphics Proceedings’.


Deussen, O., Kobbelt, L. and Tücke, P. [1995], Using simulated annealing to obtain good nodal approximations of deformable bodies, in ‘Sixth Eurographics Workshop on Simulation and Animation’, Springer.


Fisher-Cripps, A. C. [2000], Introduction to Contact Mechanics, Springer Verlag New York, Inc.


Haluck, R. S., Webster, R. W., Snyder, A. J., Melkonian, M. G., Mohler, B. J., Dise, M. L. and Lefever, A. [2001], A virtual reality surgical trainer for navigation in

Harders, M. [2006], ‘Hysteroscopy simulator’. \textsc{URL}: http://www.hystsim.ethz.ch


Kuchenbecker, K. J., Fiene, J. and Niemeyer, G. [2005], Event-based haptics and acceleration matching; Portraying and assessing the realism of contact, in ‘World Haptics Conference’, Pisa, IT.


Melerit medical [2005], ‘Pelvic vision, trauma vision’.  
**URL:** http://www.meleritmedical.com


Mendoza, C., Sundaraj, K. and Laugier, C. [2002a], Faithfull haptic feedback in medical simulators, *in ‘International Symposium in Experimental Robotics’,* Italy.

Mentice AB [2000], ‘Procedicus mist: Key surgical techniques employed in laparoscopic surgery’.

URL: http://www.mentice.com


Richard, C. [2000], On the identification and haptic display of friction, PhD thesis, Department of Mechanical Engineering, Stanford University.


Shabana, A. A. [1989], Dynamics of Multibody Systems, John Willey & Sons.


Stanislaw, H. and Todorov, N. [1999], ‘Calculation of signal detection theory measures’, Behavior Research Methods, Instruments, and Computers 1, 137–149.

Sundaraj, K., Laugier, C. and de Casson, F. B. [2003], Intra-operative ct-free ex-
anamination system for anterior cruciate ligament reconstruction, in ‘Proc. of the
IEEE-RSJ Int. Conf. on Intelligent Robots and Systems’, Las vegas, NV, USA.

Surgical Science Sweden AB [2001], ‘Lapsim: Learning basics of laparoscopy, critical
phases of laparoscopic cholecystectomy’. 
URL: http://www.surgical-science.com

Suzuki, N. and Suzuki, S. [2003], Surgery simulation system with haptic sensation
and modeling of elastic organ that reflect the patients’ anatomy, in ‘Proceedings

Székely, G., Brechbühler, C., Dual, J., Enzler, R., Hug, Hutter, R., Ironmonger,
N., Kauer, M., Meier, V., Niederer, P., Rhomberg, A., Schmid, P., Schweitzer, G.,
Thaler, M., Vuskovic, V., Tröster, G., Haller, U. and Bajka, M. [2000], ‘Virtual

Székely, G., Brechbühler, C., Hutter, R., Rhomberg, A., N.Ironmonger and Schmid,
P. [1998], Modelling of soft tissue deformation for laparoscopic surgery simula-
tion, in W. W. M. et al., eds, ‘First International Conference on Medical Image
Computing and Computer-Assisted Intervention MICCAI’98’, Lecture Notes in

Tendick, F., Downes, M., Goktekin, T., Cavusoglu, M., Feygin, D., Wu, X., Eyal,
laparoscopic surgical skills’, Presence 9(3), 236–255.

Terzopoulos, D., Platt, J., Barr, A. and Fleischer, K. [1987], Elastically deformable
models, in ‘SIGGRAPH ’87: Proceedings of the 14th annual conference on Com-
puter graphics and interactive techniques’, ACM Press, New York, NY, USA,

Teschner, M., Heidelberger, B., Müller, M. and Gross, M. [2004], A versatile and
robust model for geometrically complex deformable solids, in ‘CGI ’04: Proceed-
ings of the Computer Graphics International (CGI’04)’, IEEE Computer Society,
Washington, DC, USA, pp. 312–319.

Teschner, M., Heidelberger, B., Müller, M., Pomerantes, D. and Gross, M. H. [2003],
Optimized spatial hashing for collision detection of deformable objects., in T. Ertl,

VEST Systems [2001], ‘Medical science at your fingertips’. Informational flyer.
URL: http://www.select-it.de


Wriggers, P. [2002], *Computational Contact Mechanics*, John Willey & Sons.


Acknowledgements

At this point I would like to thank all the people, who helped and supported me during my studies and thus contributed in many different aspects to the work hidden behind these lines.

Firstly I would like to thank Prof. Dr. Gábor Székely for giving me the great opportunity of professional enrichment by accepting me for this interesting project. I also thank him for finding time for me in his busy schedule, whenever I had doubts about the future steps in my project.

I am grateful to my supervisor PD. Dr. Matthias Harders for his great help when writing this manuscript and all my other publications and for directing the experimental part of my research.

Warm thanks to my co-referee Prof. Dr. Abderrahmane Kheddar for reviewing this thesis, for his valuable suggestions and kind support.

I very happily acknowledge the collaboration with the people from the material science lab, Dr. Davide Valtorta, Dr. Alessandro Nava and Marc Hollenstein. I thank them for supplying me with the physical background on the topic of soft materials and for processing the material parameter estimation tests of my experimental silicone samples.

I am grateful to Dr. Theresa Cooke, Dr. Marc Ernst and Dr. Christian Wallraven for our fruitful collaboration, for sharing with me their knowledge on psychophysical evaluations and for their helpful insights and support when preparing joint publications.

It has been a pleasure to work with Dr. Miguel A. Otaduy. I thank him for his friendly attitude and the numerous discussions which widely broadened my knowledge on physical simulations and haptic rendering. Under his supervision I smoothly passed through the most difficult parts of my research.

Most of all, thanks must be acknowledged to Peter Čech, a devoted friend always willing to help with any technical problem, especially concerning how to govern my computer and correct my broken code; to Dr. Herbert Bay and Dr. Martina Corso for their constructive criticism on my work and for their incessant encouragement.
during my endeavours; to Dr. Gerald Bianchi and Bryn Lloyd for the profitable conversations on soft tissue modelling and their helpful advises to my ongoing research. My special thanks go to my office mates Dr. Esther Koller-Meier and Andreas Ess, the haptic team Dr. Christopher Spuhler, Dr. Alexander Barlit and Benjamin Knoerlein, and to all my colleagues at BIWI, the good fellows, who always filled me with a good mood and who converted the institute to a friendly and pleasant working place.

Finally, this thesis would not be possible without the moral support and the endless motivation of my family and my beloved fiance Martina.

This research was partially supported by the TOUCH-HapSys EU (http://www.touchhapsys.org) and the CO-ME NCCR (http://co-me.ch) project.
Curriculum Vitae

Personal data

Name Peter Leškovský
Date of birth 8. February 1979
Place of birth Bojnice, Slovak Republic
Citizenship Slovak

Education

2003 – 2007 PhD student at ETH Zurich, Department of Information Technology and Electrical Engineering, Computer Vision Lab
2002 – 2003 One semester preparation program for the doctoral studies, participant of the CGC European Graduate Program at ETH Zurich
1997 – 2002 Master studies at the Comenius University, Faculty of Mathematics, Physics and Informatics, Awarded a qualification title Master of Science in Computer Science and Informatics