Scuola Superiore di Studi Universitari e di Perfezionamento S.Anna

Laboratorio PERCRO

Multirate and Perceptual Techniques for Haptic Rendering in Virtual Environments

Ph.D. Thesis

by

Emanuele Ruffaldi

Scuola Superiore di Studi Universitari e di Perfezionamento S.Anna

Laboratorio PERCRO

Multirate and Perceptual Techniques for Haptic Rendering in Virtual Environments

PhD Thesis on Perceptual Robotics

Tutor: *Prof. Massimo Bergamasco*

Collegio dei Docenti:

Emanuele Ruffaldi

Prof. Massimo Bergamasco

Prof. Paolo Ancilotti

Asst. Prof. Carlo Alberto Avizzano

Keywords:

haptic interfaces, voxel, friction, grasping.

Emanuele Ruffaldi
Scuola Superiore di Studi e di Perfezionamento S.Anna - Laboratorio PERCRO
Piazza Martiri della Liberta 33, 56100, Pisa.
Phone: +39 050 883 287, Fax: +39 050 883 333
Email: pit@sssup.it
URL: http://www.teslacore.it

Acknowledgments:
This work has been partially done in the context of the European ENACTIVE Network of Excellence (http://www.enactivenetwork.org), reference code IST-2004-002114.

A research period at the Stanford Robotics Lab (CA, USA) was supported by a grant from the National Institutes of Health http://www.nih.gov.

Contents

1 Introduction **1**

1.1 Introduction . . . 1

1.2 Haptic Systems . . . 4

1.2.1 Tool Mediated Interaction . . . 4

1.2.2 6-DOF Haptic Rendering . . . 5

1.2.3 Direct rendering and Virtual Coupling . . . 6

1.2.4 Rigid body simulation . . . 7

1.2.5 Stability . . . 8

2 A Friction Model for Grasping **9**

2.1 Introduction . . . 9

2.1.1 Nomenclature . . . 10

2.1.2 Basic assumptions . . . 10

2.1.3 In-vivo Fingertip Models . . . 11

2.1.4 Haptic Rendering Algorithms . . . 14

2.2 Sliding Grasping Model . . . 15

2.2.1 The Friction Cone Algorithm . . . 15

2.2.2 Evaluation of grasping load and slip force in pick and place . . . 16

2.3 Soft Finger Contact Model . . . 18

2.3.1 The proxy algorithm with uncoupled friction . . . 19

2.3.2 Analysis of the grasping conditions . . . 22

2.3.3 Coupled Soft Finger Proxy Algorithm . . . 25

2.3.4 Experimental validation and applications . . . 27

2.4 Discussion . . . 30

3 Voxel Based 6DOF Haptic Rendering **35**

3.1 Introduction . . . 35

3.1.1 Collision Detection . . . 36

3.2 Voxel Collision Detection . . . 38

3.2.1 Overview . . . 39

3.2.2 Implicit Representation . . . 39

3.2.3 Collision Detection . . . 40

3.2.4 Optimizations . . . 40

3.2.5 Sensation Preserving . . . 41

3.3 Haptic Collision Response 42
3.3.1 Review of McNeely Approach 42
3.3.2 Overview 43
3.3.3 Collision Response 44
3.4 Discussion 45

4 Benchmarking Framework for 3DOF Haptic Rendering 47
4.1 Benchmarking 47
4.2 Benchmarking Framework for 3DOF 48
4.2.1 Data acquisition 48
4.2.2 Algorithm Evaluation 50
4.2.3 Results 53
4.2.4 Discussion of the results 55
4.3 Discussion 56

5 Integrating Haptic interaction on the Web 57
5.1 Rationale 57
5.1.1 Multimodal Systems on the Web 58
5.1.2 Multirate in Virtual Reality 58
5.2 Architecture 60
5.2.1 XVR 61
5.2.2 The CHAI Haptic library 62
5.2.3 Device Access 62
5.2.4 Expressing Haptics 63
5.2.5 Web Integration 64
5.2.6 Extensibility through Python - PYXVR 66
5.3 Evaluation 69
5.3.1 Haptic Loop 69
5.3.2 Force Rendering Tests 70
5.4 Applications 72
5.4.1 Haptic Pool 72
5.4.2 Virtual Restoration 73
5.5 Discussion 73

6 Summary 77

A GNU Free Documentation License 79

Bibliography 87

List of Figures

1.1 Two examples of commercial Haptic interfaces 2
1.2 The layered organization of a Haptic Systems can be described in a ways similar to the OSI Stack used for the layering of Networked applications, moving from hardware related features to the application level 3
1.3 A taxonomy of Human Robotic interaction using simple building blocks 5
1.4 Structure of the 6-DOF Haptics topics 6

2.1 Contact of a viscoelastic sphere over a plane 13
2.2 The linear friction cones 15
2.3 Example of object grasping 16
2.4 Mean grip force as a function of stiffness, friction and mass values 18
2.5 Mean grip force and as a function of stiffness, friction and mass values 19
2.6 Relative safety margin during grasping vs. manipulated mass for different condition 20
2.7 Grip vs. slip force, showing the relative safety margin during grasping of virtual objects 21
2.8 Soft Finger Proxy for the grasping of virtual objects 21
2.9 The classical friction cone for simulation of linear friction 22
2.10 The plane r, ρ under the not coupled algorithm 23
2.11 Condition between sliding and rotational. With the experimental mr and ms . . . 24
2.12 Plane r,ρ with the sliding and rotation conditions 25
2.13 The interpretation of the mixed rotational-linear friction adaptive cone 26
2.14 Condition between sliding and rotational. With the therical mr and ms 26
2.15 The plane r, ρ under the coupled algorithm 27
2.16 The coupled region with the condition that prevents static behavior on the left, and the static region on the right 27
2.17 Rotation and Slide using Coupled 28
2.18 Manipulating virtual objects using two fingers per hand 28
2.19 The movement of a rectangle block grasped among two fingers under condition A 29
2.20 Trajectory vs. time (x and y of CM and rotation α around GP) of the simulated motion under condition A 30
2.21 Representation of the repositioning of the god-object in the $[r, \rho]$ plane under condition A 31
2.22 The movement of a rectangle block grasped among two fingers under condition B 32

2.23 Trajectory vs. time (x and y of CM and rotation α around GP) of the simulated motion under condition B 32
2.24 Schematic representation of the experiment 33
2.25 Box Plot of the alignment error with different lengths and the two algorithms . . . 33

3.1 Selected characteristics for the new algorithm 39
3.2 Example of sphere tighting depending on the number of children in the quadtree 41
3.3 When an octree has a single child the collision detection can skip one level 41
3.4 Various cases for the computation of the voxel support area 42
3.5 Tagent Plane Force Model 43
3.6 Separating and Incoming Contact Pairs 44
3.7 Example of collision response performed by two steps of the algorithm. The black dots are the collision pairs obtained from the voxel collision detection. The arrows represent the impulses computed at each step. 45
3.8 A snapshot of the collision detection and response of the algorithm with two impulses generated for the resoltution 45

4.1 The sensor used to acquire force and torque information, alongside a coin to indicate scale 49
4.2 Our data acquisition system couples a custom handle and a small scanning probe with a force/torque sensor 50
4.3 An overview of our data processing and algorithm evaluation pipeline. An object is scanned, producing a 3D geometric model and an out-trajectory. An in-trajectory is synthesized from this out-trajectory and is fed as input to a haptic rendering system, which produces force information and (for most algorithms) a new out-trajectory, which can be compared to the physical scanning data. 51
4.4 An "out-trajectory" represents the path taken by a physical probe over the surface of an object; a haptic rendering algorithm typically approximates this trajectory with an "in-trajectory" that allows the probe to enter the virtual object. 51
4.5 Computation of an in-trajectory point from a sampled out-trajectory point. . . . 52
4.6 Our evaluation approach is able to identify and quantify failure cases for the Proxy algorithm 55

5.1 Multiple representations of the same object 59
5.2 Schematic representation of a multimodal entity and its connection to other entities 59
5.3 The architectural view of HapticWeb 60
5.4 A picture that shows an interactive session with HapticWeb 61
5.5 This diagram shows the loops of a typical multimodal system withing XVR 62
5.6 A schematic representation of haptic effects provided by HapticWeb, in the top row the field effects and in the bottom row constraint effects 64
5.7 An example of integration of the HapticWeb system inside a Web page for new types of documents 66
5.8 Architecture of the PYXVR system, showing the relationship between the two scripting systems and the modules 67

5.9 The three ways of implementing the graphic-haptic loops: (a) native only (b) script only (c) intermediate based. Each circle represents a functional loop and the oscillation gives a qualitative idea of the loop's rate 70
5.10 This is an example of the probed trajectory used for benchmarking the haptic interaction with the model using the various haptic loop approaches 71
5.11 The graph shows the relation between the haptic loop period and the graphic load, comparing the native case (1ms) against the script-based loops with different resolutions of the model 71
5.12 Example of the Haptic Pool application in which the GRAB device is being used . 72
5.13 A sequence of snapshots of the pooling demo application 75
5.14 Possibility of adding spinning effects while hitting the ball 76
5.15 Virtual Restoration application, working with the two arms of the GRAB device . 76

List of Tables

2.1 Differences among 4 investigated models 11
2.2 Correlation table ($^{**}p < 0.001$). 17
2.3 Experimental values of means $\pm SD$ of μ_l, μ_r and their ratio measured at the index fingertip for different materials 29

4.1 Results obtained from an analysis of haptic rendering using a Proxy algorithm on a series of progressively more refined synthetic spheres 54
4.2 Results obtained from an analysis of haptic rendering using a Proxy algorithm on a series of progressively more refined planar meshes 54
4.3 Comparison of several algorithms processing the geometry illustrated in Figure 4.6 54

Thanks

Three years have been passed since the beginning of this PhD. It has been a part of my life in which I've studied and worked for improving my understanding of Virtual Environments and for contributing to the society. Such period would be nothing without the people with which I've worked, or that I've provided insightful help to my work.

First of all I would like to thank my professor Massimo Bergamasco, that has not only guided and inspired me on the research but also transmitted me an interest on certain aspects of design and art that I consider extremely valuable for the future. Almost nothing of this work could be completed without the enjoying and brillant support provided by Antonio Frisoli and Carlo Alberto Avizzano guiding me into new topics or helping me focus on existing work.

All the people of PERCRO have contributed to this PhD, and in particular I thank everyone with which I've enjoyed the abroad experiences, in which work and fun were mixed in an amazing way.

In my last part of this PhD I've spent a period in the BioRobotics lab at Stanford University, a good opportunity for understanding new topics and having a different view on the research. I would like to thank prof. Kenneth Salisbury and prof. Sabine Girod for the opportunity they gave me and the help during such period. This experience anyway would be not valuable without the help of Federico Barbagli that has supported and inspired me for this time and helped shaping my work. I would like to thank all the people of the BioRobotic lab in particular Dan and my friend Nicola.

A special thank, that comes from the heart, goes to Elisabetta, the woman of my life, that has helped and inspired me in this Thesis, always showing the positive side of life.

To my family and Elisabetta

Chapter 1

Introduction

This chapter introduces the topics addressed by the Thesis presenting the basic principles of Haptic interaction in Virtual Environments

1.1 Introduction

The exploration and interaction with the World comes through various human senses. Among them vision and sound are predominant because of their range in frequency, spatial capabilities and complexity. The sense of touch instead is a different perceptual channel that is local to the human and in which the spatial component is mapped over the whole human body. This locality has been remarked by [100] calling touch a “reality sense” because touch is not externally mediated as vision or sound. The role of touch and the external environment is studied by a discipline named Haptics, also defined by Gibson [48] as *the sensibility of the individual to the world adjacent to his body by the use of his body*.

The advancements of Medicine, Robotics and Computer Science have not only improved the understanding of these perceptual channels but also reached a technological level that allows the simulation of these senses. A Virtual Environment is a computer generated environment in which a human being is able to interact and perceive as in a real environment, although with various levels of realism. The human interaction with the Virtual Environment is provided through Multimodal Interfaces, that are a general Human Computer Interface able to stimulate the different perceptual channels. Haptic Interfaces are robotic systems that allow the simulation of the sense of touch focusing on two main aspects of Haptic interaction: kinesthetic and tactile. The perception of macroscopic forces over the human body is covered by kinesthetic interaction while the perception of surface properties stimulated over the skin is represented by tactile interaction. These two kinds of feedback have different perceptual and technological characteristics, and this Thesis focuses on kinesthetic interaction.

The typical haptic interface is able to provide a single force feedback applied on a single point of the human body, eventually associated with a torque. We measure the capabilities of such haptic interfaces by the number of active Degree of Freedoms (DOF). The simplest haptic interface is a wheel that is able to exert force against the user, in this case we say that it is a 1-DOF interface. A planar haptic interface, like a pantograph is able to exert forces only along the two horizontal axes of the plane, and it is a 2-DOF interface. A typical commercial haptic interface has a stylus that is held in the user hand and it is able to exert the forces along the three directions, it is a 3-DOF interface. Finally a more realistic interaction is provided by a 6-DOF interface that is able to exert torques over the user, although at the cost of more complexity. We characterize also the haptic interfaces from the number of contact points, that correspond to the number of points that are able to exert forces to the user. With two contact points, although of 3-DOF each, is possible to simulate the grasping and the manipulation of objects. In this work we are interested in single point of contact interfaces with 3-DOF and 6-DOF. Figure 1.1 shows on the right one of the most used commercial Haptic Interfaces a PHANToM Premium [92] by Sensable Inc a that allows one point 3-DOF interaction. On the left instead there is the CyberGrasph by Immersioon [69], an example of 1-DOF five point of contact interface that can be used to simulate the full hand grasping of objects.

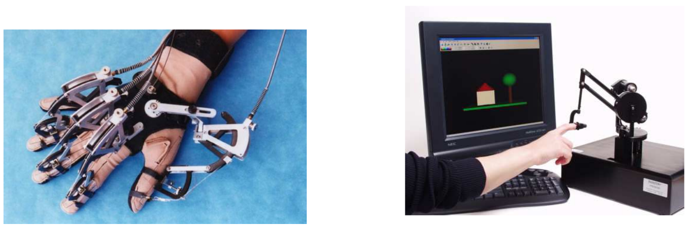

(a) A Cybergrasp Glove (b) The PHANToM Omni

Figure 1.1: Two examples of commercial Haptic interfaces

The research on Haptics involves various disciplines from neurosciences to mechanics, and among them Computer Haptics [136] is the one that covers research on software aspects. A complete Haptic System requires software from the low level control of the Haptic Interface, to the high level simulation of haptic interaction in the Virtual Environment. The range of aspects of a Haptic System can be described using a stack representation that has similarities to the OSI Network Stack, as shown in Figure 1.2.

The ideas and the understandings obtained during this research activity are presented in this Thesis:

The set of algorithms and software that compute the force feedback given the user interaction with the haptic interface is called Haptic Rendering (HR) [124], a name that is taken from the world of Computer Graphics (in which rendering is the operation of presenting a visual representation of a geometrical and mathematical world stored in a computer. Haptic Rendering has some characteristics in common with the graphical rendering, but it is different not only in the higher refresh rate required by humans for this perceptual channel, but also in the fact that the final result is a force vector and a torque for each point of contact, an aspect that is extremely different from the millions of points generated for a graphic display. If in the graphical display is necessary to take into account the capabilities of the human vision channel, in the case of Haptics more work is needed because the topic is younger and it has higher refresh

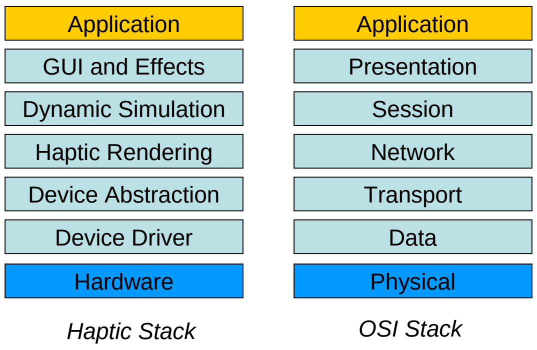


Figure 1.2: The layered organization of a Haptic Systems can be described in a ways similar to the OSI Stack used for the layering of Networked applications, moving from hardware related features to the application level

requirements.

In the context of Haptic Systems the research activity on which this Thesis is based has focused on a multirate-perceptual approach aimed at the improvement of the haptic interaction both in terms of realism and development tools. The vision is the possibility to create basic building blocks for the creation of haptic enabled application that can be integrated by developers without dealing with detailed perceptual and implementation aspects. The haptic manipulation of objects is addressed in the first part of the Thesis presenting two new algorithms for grasping and 6-DOF manipulation. Such design has been followed by a benchmarking methodology that can be applied to 3-DOF algorithm and that could be extended to 6-DOF. Finally an overall development framework for Haptic application is presented in the last chapter.

- **Chapter 2** presents the first contribution to Haptic Rendering with a soft-finger proxy algorithm that allows the grasping of objects in a virtual environment. The algorithm has been evaluated with a two arm 3-DOF haptic interface and implemented using a multi-rate application framework that allows the integration of haptic rendering and dynamic simulation.

- The research toward the realism of Haptic Rendering has been continued by a work on 6-DOF rendering based on Volume Models presented in **chapter 3**. This work provides a general tool for haptic interaction that can be efficiently formed with volumes models obtained from polygonal representations or from medical imaging.

- The design and implementation of Haptic Rendering algorithms is always delicate because of both perceptual and performance related aspects. The **chapter 4** focuses on a Benchmarking framework that has been applied for the evaluation of 3-DOF algorithms.

- The overall vision relative to the Haptic pipeline is completed by a framework for the fast development of Haptic applications. This framework, called HapticWeb, is described in detail in **chapter 5**.

- Finally **chapter 6** summarizes the vision and the research presented in this Thesis with some considerations relative to the future.

1.2 Haptic Systems

Haptic Interfaces find their origin in the development of telerobotic systems. A telerobotic systems is a kind of robotic system in which one side, the master, is manipulated by the user, and the other, the slave, performs actions depending on the movements of the master, as originally proposed by [51]. A key problem of teleoperated system is the difference in response and behavior respect the direct manipulation of the slave tool, a problem that is addressed by designing *transparent* systems. The first step toward Haptic Interfaces were the creation of a telerobotic master based on cartesian control and enough flexible to be adapted to different kinds of slaves [12]. When the slave is replaced by a computer generated simulation we obtain a Haptic Interface, that allows a complete simulation of the slave behaviors.

The Figure 1.3 presents a simple taxonomy of Human Machine interaction based on a simple grammar of entities, as Human, Robot, Computer and Network. In this diagram the first element is the standard case of a computer controlled robot, followed by the classic teleoperated system, eventually extended from to Networked teleoperation. The evolution to Haptic Interface comes by the replacement of the slave with a Virtual Environment.

Haptic Interfaces in Virtual Environments has been extensively used in many applications, from cultural heritage, to medicine, to industrial design and chemstry. Most applications use a Desktop Haptic Interfaces, like the PHANToM [92], but devices with higher performances and flexibility has been used for specific applications and in immersive Virtual Environments [13, 40, 16].

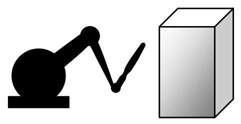
A Haptic Interface is also a mean of communication when integrated in a Collaborative Virtual Environment (CVE). The fifth case of Figure 1.3 shows a collaborative haptic environment in which two or more user operate in the same Virtual Environment [113, 37].

This section presents Haptic System focusing on the various approaches to Haptic Rendering, first addressing the general concept of tool mediated interaction and 3-DOF rendering, then presenting the 6-DOF rendering taking into account force generation, dynamic simulation and stability.

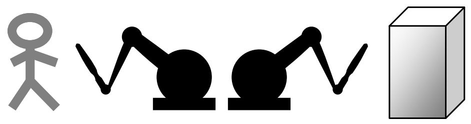
1.2.1 Tool Mediated Interaction

The haptic interaction that the user experiences in a Virtual Environment is mostly tool mediated, because of the limited number of contact points that are available. The user physically interacts with the haptic interface and controls a Virtual Body that is present in the Virtual Environment. In the case of a 3-DOF interface the feedback provided by the tool is limited to a single force feedback, and it corresponds to the force applied to the tip of such tool. For this reason the tools associated to 3-DOF applications are mostly stylus based.

The Haptic Rendering of such 3-DOF tools involves the computation of the force feedback only on the point of the tip, eventually represented by a small sphere. The point used in 3-DOF rendering is directly attached to the haptic interface and it is able to move in every space, and when it get inside a virtual object a force is applied to simulate the contact surface. Such force is computed using a simple geometrical algorithm that keeps track of a point always on the surface called proxy and the force vector is directed along the distance between the proxy point and the tip point inside the object, with a modulus that depends on the penetration depth and a

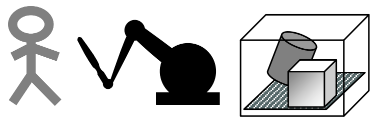


(a) A robot

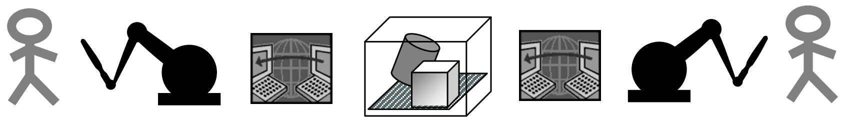


(b) A locally teleoperated system with a master and slave

(c) A networked teleoperated system



(d) A haptic interface used inside a Virtual Environment



(e) A multi user collaborative Virtual Environment with haptic interaction

Figure 1.3: A taxonomy of Human Robotic interaction using simple building blocks

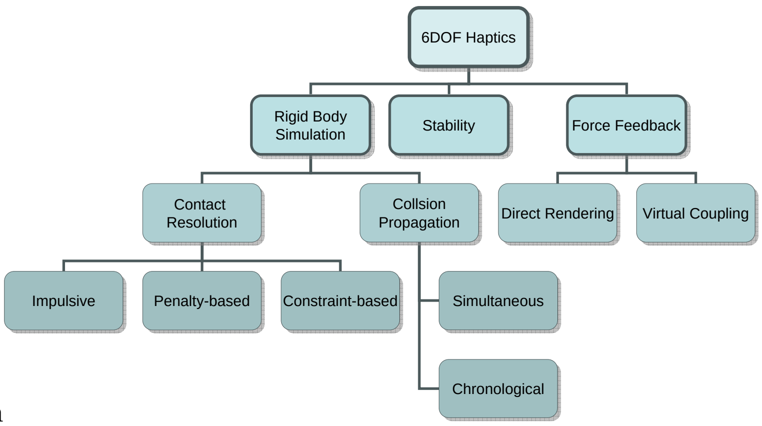
proportional coefficient that is used to simulate the stiffness of the surface [147, 122]. Additional force contributions can be added to better simulate the first contact of the tip with the surface, and also to simulate haptic surface properties like friction and textures.

The above 3-DOF approach has been improved in literature by taking into account another important aspect of touch, the contact event. In a real contact event the force feedback is not proportional to the penetration depth, almost zero in the proxy approach, but the contact produces an impuslive force that depends on the material and the gripping of the user hand over the probing device [82],[45].

1.2.2 6-DOF Haptic Rendering

The complexity of the interaction with 6-DOF Haptic Rendering is required when dealing with tasks in which the torque feedback is fundamental for the completion of the task itself. For example the case of manipulation of mechanical parts and the evaluation of the their assembly, called Virtual Prototyping, is difficult to be achieved successfully using 3-DOF interfaces. The 6-DOF Haptic Rendering involves the full simulation of the tool's body and dynamics for the correct computation of the resulting torque and force feedback. In 6-DOF Haptic Rendering the single point of the 3-DOF is replaced by a complex geometry that depends on the application and that is connected to the haptic point directly or by a damped 6-DOF spring. The simulation in 6-DOF HR requires first the computation of the Collision Detection between the body and the other objects in the Virtual Environment and then a Collision Response for transmitting to the user the sensation of the collision. Many advanced Haptic applications and tasks can be performed with 3-DOF feedback without involving the complexity of the 6-DOF interfaces and algorithms.

The fundamental problem in 6-DOF haptics is represented by the real time response of the collision computation between the probe and the bodies. When is not possible to perform all the phases of the Haptic Rendering pipeline at the same haptic rate, it is necessary to adopt a multirate approach in which different components are executed at different rates and their execution is synchronized ([126]). An additional variant of this approach is the use of an intermediate representation or local model, in which the high rate haptic rendering is computed on a local representation of the body near the haptic contact point. [4],[35]. The structure of this topic is presented in Figure 1.4.



cmcm

Figure 1.4: Structure of the 6-DOF Haptics topics

1.2.3 Direct rendering and Virtual Coupling

6-DOF haptic rendering algorithms can be organized in two main categories depending on the way they relate the position of the haptic probe object with the position of the haptic interface point. The first approach is the direct rendering in which the position of the haptic probe object is matched with the position of the haptic interface point in the environment. In this way the control of the probe is direct and without delays. The side effect of this approach is in large penetration depth and instabilities caused by lower frame when the collision detection algorithm slows down. ([77],[2],[75],[104]).

The other approach uses the concept of Virtual Coupling ([31]) in which the haptic rendering computes a dynamic simulation of the haptic probe and the user controls the object by a bidirectional spring that connects the haptic interface point and the probe object. (constrained:

[15], [121],impulsive: [24],[33], penalty: [144] [89],[96]).This solution provides a much more stable response but it has the effect of producing a smoothing of the feedback. In some way the Virtual Coupling used for 6-DOF haptic can be considered as a local model in which the intermediate representation is a spherical force field.

The Virtual Coupling defines a coupling frame of reference with position $\mathbf{x}_c$ and rotation $\mathbf{q}_c$ computed from the dynamic simulation and expressed in body coordinates. The coupling frame is connected to the haptic device interaction point ($\mathbf{x}_h$, $\mathbf{q}_h$) by a viscoelastic link that affects the interaction through a coupling force $\mathbf{f}_c$ and torque τ_c. The Virtual Coupling is defined by the four parameters of linear-rotational stiffness and damping, plus the mass of the grasped object. The following is an example of formula that expresses the virtual copluing forces and torques:

$$F_c = k_c(x_h - x - Rx_c) + b_c(v_h - v - \omega \times x_c) \tag{1.1}$$

$$T_c = (Rx_c) \times F_c + k_\theta u_c + b_\theta(\omega_h - \omega) \tag{1.2}$$

Where $k_c, b_c, b_\theta, k_\theta$ are the coefficients, and u_c is the rotation axis between the two rotation frames q_h and q_c.

1.2.4 Rigid body simulation

There are several methods for organizing the dynamic simulation used in the Virtual Coupling: penatly-based, contraint-based and impulse based. These methods, typically used in rigid body dynamic simulation, can be applied to haptic simulation depending on their integration time requirements.

Penalty-based methods identify two contact states, non contact and contact and they respond to the contact state with a force that is proprotional to the penetration depth and the stiffness of the materials. They are suitable for haptics because they are efficient but limits in the integration step impact on the damping and in the perceived stiffness of the bodies [2],[32],[101],[96], ([144] [89] [94]). An interesting approach described in [61] uses the volumetric penetration instead of the depth for handling the case of face-face contacts. For non haptic works on the topic see [93],[76],[138].

Constraint-based methods identify three contact states: non contact, collision contact and resting contact. The integration is performed between non colliding contact events, and the resting contacts are described as constraints. When a collision event occurs the integrator is reset to the contact time and impulse forces are applied to prevent the penetration. For the application of these methods to haptic interaction the problem of variable integration time needs to be addressed and also the the constraints used in the simulation are replaced by a penalty-based approach at the level of the haptic controller [148],([122],[120],[121]),[15]. In the general case of dynamic simulation constraint methods are grouped in the analytical methods [6],[8].

Finally impulse-based methods identify two contact states: non contact and colliding contact. These methods respond to colliding contacts with a force impulse and to resting contacts with a sequence of micro impulses [98]. When applied to haptics this approach has problems in dealing with resting contacts and dry friction [24],[33].

In general the main problem is the response to the first contact that should generate a large force response to simulate the contact impact. This problem has been expliclity addressed by the use of braking forces [123],[144], or by an open loop response as in the case of event based haptics [67],[82],[45]. A general solution proposed by [34] is to use a hybrid approach of force pulses at the initial contact and then to use a penalty based response for the resting contact.

Another aspect of the simulation of rigid bodies is the handling of the contact area. Most of the algorithms deal with point contacts but they need to take use some special handling for face-face collisions. For implicit curves [133], volumetric penalty depth [61], point sampling [138].

1.2.5 Stability

The *stability* of haptic interaction is a fundamental aspect because it affects the quality of the feedback. Research has focused on virtual stiff walls that are the basic building blocks of haptic interaction. The first requirement that came out for providing such stiff walls was a high update rate but it is not sufficient for providing stability. When considering the overall system of the Human, the Haptic Interface, the Haptic Control Loop , the Haptic Rendering algorithm and the Virtual Environment the simple haptic rate factor is not more adeguate for the guarantee of the stability.

The relationship between the different system components can be evaluated in terms of energy transfer and the resulting analysis provides a better insight about the stability. Energy based approaches have identified the passivity of the system as a sufficient requirement for the stability as long as the user is considered a passive entity [30],[50],[71],[64],[90]. In reality the user is not a passive entity and also there are non idealities caused by the hardware components that tend to introduce energy in the system.

The evaluation of the stability of a haptic system is typically measured through user experiments. In this case is better to refer to it as user *perceived instability*. Various works by Choi and Tan [28], [27],[26] measured perceived instability caused by the haptic rate and the use of haptic textures.

When considering 6-DOF haptic rendering the energetic requirement for a stable interaction is to not introduce new energy during the collision response, a concept that has recently addressed by [34].

Chapter 2

A Friction Model for Grasping

This chapter discusses about a friction model and a new soft finger proxy algorithm that allows to provide a realistic grasping interaction with virtual objects using a two arm 3-DOF device.

2.1 Introduction

One of the key features of human fingertips is to be able to resist moments, up to a torsional friction limit, about contact normals. This simple, and yet essential feature, allows humans to fully restrain objects using two fingertips, something that would be impossible to do using the tip of two tools. As new haptic devices allowing interaction through multiple points of contact are being created [140, 69, 5, 9], it is essential to be able to simulate this type of contact in order to support tasks such as virtual grasping. Haptic rendering algorithms simulating point-contact, such as the proxy [122] and the god-object [147], have been popular for a decade thanks to their computational efficiency, but fail to model the rotational friction capabilities of the human fingertip. More complex haptic rendering algorithms [144] may allow users to simulate virtual manipulation tasks but are more computationally expensive. Additionally neither class of algorithms have been tuned to specifically simulate human fingertips. An exception to this is the recent [25] that uses a set of spheres to better simulate the friction based interaction of the finger with virtual objects, and also [39].

This chapter presents first a general overview on the fingertip properties for the grasping, the possible mathematical models and it shows a new soft finger proxy algorithm that allows to simulate the grasping of virtual objects using a two arm 3-DOF device. Suche algorithm has been evaluated with users first simulating only the sliding of the object and then both the sliding and the rotation of the object defining a coupling between the two constraints.

The following are the conventions adopted in the rest of the chapter:

2.1.1 Nomenclature

a	Radius of the contact area on the fingerpad
$p(r)$	Pressure distribution law over the contact area
P	Total normal force applied over the contact area
M	Friction moment induced by normal force P
δ	Distance between point of application of P and center of contact area
r	Distance between generic point and center of the contact area
q	Tangential traction forces over the contact area
F_{fr}	Maximum component of tangential traction force due to static friction
r_m	Equivalent radius arm of the tangential friction distribution
μ	Static and dynamic linear friction coefficient
μ_r	Rotational friction coefficient
$\Gamma(P)$	Analytical relationship between M and P
D	Length of the rectangular shaped object
L	Arm of the momentum applied by the center of mass respect the rotation axis

2.1.2 Basic assumptions

Some basic assumptions, typically adopted by existing literature on human fingerpad modelling, are maintained in this work. The fingerpad is modelled as a sphere, the contact area is assumed to be a circle of radius a, and the pressure distribution is assumed to be axial-symmetric.

Under the effect of contact force P, a distribution of pressure $p(r)$ is generated over the contact area, such that:

$$P = \int_0^a p(r) 2\pi r dr \tag{2.1}$$

Under static conditions, friction forces depend on the friction coefficient μ. In such case p produces on a infinitesimal area dA a tangential traction q such that:

$$|q| \leq \mu p \, \mathrm{dA} = F_{fr} \tag{2.2}$$

The local values of F_{fr} determine the conditions for which slip between the two bodies in contact can occur, and generate a friction moment M given by:

$$M = \int_0^a \mu p(r) 2\pi r^2 dr = P r_m(a) \tag{2.3}$$

with

$$r_m(a) = \frac{\int_0^a \mu p(r) r^2 dr}{\int_0^a p(r) r dr} \tag{2.4}$$

Equations (2.1) and (2.3) are assumed to hold independently of the mathematical model adopted for the fingerpad.

	constitutive model	strain	stress	pressure distribution
MH/CH	half-space linear	infinitesimal	3D	elliptic
LFM	generic shell	finite	2D	uniform
VS	Kelvin model	finite	1D	quadratic

Table 2.1: Differences among 4 investigated models

2.1.3 In-vivo Fingertip Models

The characterization of human fingertips properties has been widely addressed in the past two decades by the bio-mechanics and neuroscience communities. However, model simulating the force-indentation and force-contact area behavior of the human fingertip have received most of the attention [134, 59, 135, 130, 129, 111, 110].

The study of frictional properties of human fingertips was addressed by the Neuroscience community in order to evaluate what are the minimal forces applied by humans in order to stably grasp objects [145, 84] and to resist tangential torques, i.e. to restrain objects from rotating using rotational friction [79, 52]. The main results of such researches is that in both cases the normal force that is applied by subject in order to resist tangential force and torques is always the minimal amount that ensures avoiding slippage. Moreover the ratio between normal force and tangential torque, and normal force and tangential forces is always linear.

Four possible different analytical models that simulate the normal force-frictional torque behavior of the human fingerpad are presented in the following. All four are based on pre-existing models of normal force-displacement and normal force-contact area presented in the past in the bio-mechanics and robotics community. The initial models were selected, amongst many, because of their analytical formulation, which makes them feasible for real-time applications, and because of their mostly static nature.

The first two models (Classic Hertz - CH, Modified Hertz - MH) are based on Hertzian theory (see [18] for CH and [111, 110] for MH). The third model (Viscous Sphere - VS), which was originally used to describe the behavior of plantar soft tissue, is based on a viscous sphere representation (see [60]) which can be seen as an extension of the “waterbed” model proposed by Srinivasan in [134]. The fourth model (Liquid Filled Membrane - LFM) describes the fingerpad as a fluid filled membrane (see [129]).

The proposed models’ capability to closely simulate the human fingertip normal displacement, contact area and rotational friction behavior in relationship to a given normal force P, has been evaluated in [10] using an experimental data set.

Here we report the results from the analysis of the above models using the experimental data. The four models feature different constitutive equation and thus different rotational friction properties. Table 2.1 summarizes the main differences among them.

The relationship between normal force and normal displacement has been modelled through an exponential formulation according to the experimental results found by Howe et al. [111]:

$$P = T^e(\delta) = \frac{b}{m}(e^{m(\delta-\delta_0)} - 1) \tag{2.5}$$

where $\delta_0 = 0\,\text{mm}$, $b = 0.19\,\text{N/mm}$ $m = 2.1\,\text{mm}^{-1}$.

In the non-linear range(0-2 N), characteristic of the fingerpad indentation, an equivalent power formulation of (2.5) representing the best least squares fit is found as follows:

$$P(\delta) = p_1\delta^{p_2} \quad p_1 = 0.5118\frac{\text{N}}{\text{mm}^{p_2}},\ p_2 = 3.4897 \tag{2.6}$$

CH - Classic Hertzian model

For the CH model of a fingerpad, which was determined by Brock et al. in [18], contact with a given surface is approximated as one between two elastic solids, and thus can be described using Hertzian theory [63].

The geometric constrain equation for the CH model is given by:

$$a^2 = R\delta \tag{2.7}$$

The resultant distribution of pressure over the contact area has an elliptical shape described by

$$p = p_0[1 - (\frac{r}{a})^2]^{1/2} \tag{2.8}$$

By using (2.1) and (2.3) the expressions for the law *Expr2* are found as:

$$P = \int_0^a p(r)2\pi r dr = \frac{2}{3}p_o\pi a^2 \tag{2.9}$$

$$M = \int_0^a \mu p(r)2\pi r^2 dr = \frac{1}{8}\mu p_0 a^3\pi^2 \tag{2.10}$$

and thus

$$\frac{M}{P} = \frac{3\pi}{16}\mu a(P) \tag{2.11}$$

By inspecting (2.11), it is clear that the relationship between P and M only depends on $a(P)$, that according to the classic Hertz theory is given by:

$$a = \left(\frac{3PR}{4E^*}\right)^{1/3} \tag{2.12}$$

Thus, by substituting expression (2.12) in (2.11), the law $\Gamma = M(P)$ for the Classic Hertz (CH) model is obtained:

$$M = \frac{3\pi}{16}\mu\left(\frac{3R}{4E^*}\right)^{1/3} P^{4/3} = \mu'_m P^{1.3333} \tag{2.13}$$

MH - Modified Hertzian model

Howe et al. [111] proposed a modification of the classic Hertz model to fit the experimental indentation displacement vs. force Γ (2.11). To do so, two corrective terms, the experimental instantaneous response T_e and the relaxation response, were introduced. For our purposes, if relaxation effects are neglected, we can modify expression (2.8) of p to only include T_e, and thus obtain:

$$p'(r) = p(r)T^e(\delta) \tag{2.14}$$

It can be shown that under this hypothesis the expression of the ratio M/P remains equal to (2.11).

From (2.7) and (2.6), it then follows that

$$P(a) = \frac{p_1}{R^{3.4897}} a^{6.9795} \tag{2.15}$$

and thus combining (2.15) and (2.11), the Modified Hertz (MH) model law is obtained as

$$M = \mu'_m P^{1.1433} \tag{2.16}$$

VS - Viscous sphere model

The formulation of the hertzian model given in equation (2.7) is valid only for infinitesimal deformations. When finite deformations are taken into account, i.e. when the entity of displacement due to the contact deformation is not negligible with respect to the nominal dimension of the fingerpad, this assumption is no longer valid. In order to take into account finite deformations, in the following we propose a model inspired by ideas presented in [134] and [60].

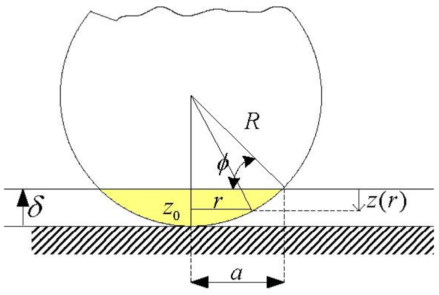


Figure 2.1: Contact of a viscoelastic sphere over a plane

Referring to Figure 2.1, we will assume that, during the contact with a plane, points lying on the fingerpad surface will be displaced of a quantity equal to their distance from the plane in the un-deformed configuration. The geometric constrain equation for this model becomes so:

$$(R-\delta)^2 + a^2 = R^2 \tag{2.17}$$

The displacement $z(r)$ is given by the following expression:

$$z(r) = \sqrt{R^2 - r^2} - \sqrt{R^2 - a^2} = \sqrt{R^2 - r^2} - R + \delta \tag{2.18}$$

We will assume that the contact pressure is modelled as a set of springs normal to the contact area, i.e. by $p(r) = kz(r)$, with $k = k'\delta^n$. The associated contact force and the friction moment are given by:

$$P = \int_0^a p(r)2\pi r dr = \frac{1}{3}\pi k\delta^2(3R-\delta) \tag{2.19}$$

$$M = \int_0^a \mu p(r) 2\pi r^2 dr = \mu\pi k[(R-\delta)a(R^2/4 - 2/3a^2) - a/2(R^2-a^2)^{3/2} + R^4/4\phi)] \quad (2.20)$$

The values of k' and n, which are unknown, have been determined as the ones which provides the best least squares fit of (2.19) with the experimental law (2.5) $P(\delta)$. The power formulation that gives best least squares fit of the law $M(P)$ in the non-linear range (0-2 N) can be derived for the VS model and is given by:

$$M = \mu'_m P^{1.1289} \quad (2.21)$$

LFM - Liquid filled membrane

Serina et al. [129] adopted a structural model for fingertip pulp based on the theory of elastic membranes [54]. While this model allows the representation of large strain deformations, it has the limitation of assuming a uniform distribution of pressure. Note that the LFM is used to model both $a(P)$ and $P(\delta)$ laws [129]. In this case P and M are given by

$$P = \int_0^a p 2\pi r dr = \pi p_0 a^2 \quad (2.22)$$

$$M = \int_0^a \mu p(r) 2\pi r^2 dr = \frac{2}{3}\mu\pi p_0 a^3 \quad (2.23)$$

and thus

$$\frac{M}{P} = \frac{2}{3}\mu a(P) \quad (2.24)$$

2.1.4 Haptic Rendering Algorithms

Three types of point contacts have traditionally been considered by the grasping community [125, 105]. A *point contact without friction* can only exert a 1-system of wrenches[1] on an object (a force along the contact normal). A *point contact with friction* can exert a 3-system of wrenches on an object (three independent forces through the point of contact). A *soft finger contact* behaves like a point contact with friction, except that its contact area is large enough that it can support moments (up to a torsional friction limit) about the contact normal.

Haptic simulation of grasping lacks in realism if users cannot completely restrain virtual objects while manipulating them as normally happens in real life. Thus creating an interface that is capable of simulating form closure [2] is a key issue. As proposed in [9] the coupling of two soft-finger contacts is the minimal configuration[3] that ensures form closure. Thus in what follows we will propose a soft-finger proxy algorithm.

It is important to note that friction models to be used in haptic simulation have been proposed in the past by various research groups (see [147, 122, 42, 103] amongst others).

[1] We use the terms wrench and twist to signify generalized forces and motions, respectively.
[2] Which can be defined as the capacity of a certain grasp to completely restrain an object against any disturbance wrench
[3] From a number of actuators/sensors perspective

2.2 Sliding Grasping Model

The haptic rendering algorithm is enhanced with a linear friction model that provides additional touch realism and is fundamental for the grasping model. The standard proxy algorithm is modified with the linear friction algorithm using the friction cone model [95]: the movement of the proxy toward of the goal is prevented by the friction itself, and perceived by the user as a tangential force. The algorithm works by building a friction cone with the top at the haptic contact point, the base centered at the god point and an aperture depending on the friction coefficient. If the last proxy position is inside the cone the proxy should not be moved and the user perceive a tangential force opposite to the moving direction, otherwise the proxy is positioned on the border of the cone. The evaluation of the position of the proxy respect to the friction cone is equivalent to decomposing the contact force in the normal and tangential components and evaluating if the tangential is greater than the normal force multiplied by the static friction coefficient. The advantage of this algorithm respect to other is only position based and it does not use a velocity estimation for the evaluation of the friction force.

2.2.1 The Friction Cone Algorithm

The static friction is extended with the dynamic friction by using a proxy algorithm with two states, slip and not slip. During the not slip state the proxy is not moved if inside the static friction cone otherwise the state is changed to slip and the proxy moved to the border of the dynamic friction cone; when in slip mode the state is changed to not slip if the proxy is inside the dynamic cone otherwise the state is kept and the proxy moved to the border of the dynamic friction. The Figure 2.2 shows the two friction cones and the transition diagram.

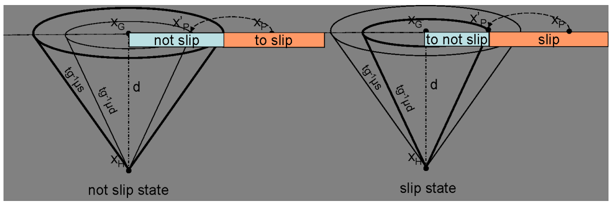


Figure 2.2: The linear friction cones

The grasping of objects in this system is based on the friction and on the dynamic simulation of the objects. The grasping force exerted by the user over the object produces a friction tangential force for each of the contact points that allows to raise and manipulate the object. This model is well integrated in the dynamic simulator because when the object becomes in contact to other objects the user receives a force feedback as a change of depth caused by the movement of the object. The Figure 2.3 shows a simple example of object grasping in this system, with the display of the two contact points and the contact forces shown as the yellow vectors.

In the design phase of the grasping there are some parameters that are correlated and should be correctly evaluated for allowing a precise grasp of the object for the application, they are the static friction coefficient, the mass of the object and the stiffness of the object itself. In the evaluation part the effect of these different parameters are analyzed on a group of subjects.

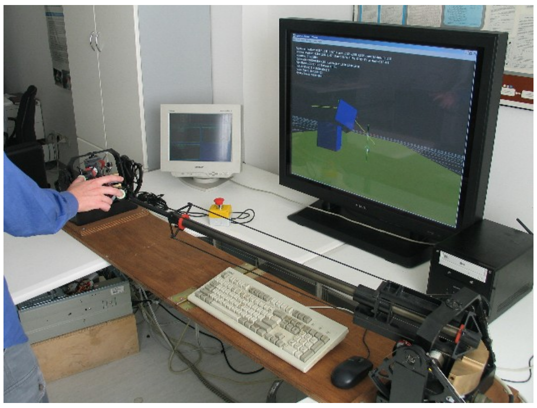


Figure 2.3: Example of object grasping

2.2.2 Evaluation of grasping load and slip force in pick and place

In order to assess the efficiency of the device and the rendering for pick and place operations, a specific evaluation was performed to compare the available data on human grasping of real objects [47, 73] with the virtual case. The influence of weight on static grip has been experimentally studied by [47], where safety margins for grasping for prevention of slipping are analyzed. The safety margin is defined as the difference between the grip force and the slip force, that is the minimum grip force required for preventing slipping.

Subjects and general procedures

Three healthy right-handed men, aged between 27 and 35 yrs, served as subjects for the study. The subjects sat on a height-adjustable chair. In this position the subject might held with his right hand the two thimbles connected to the haptic interfaces, and respectively wear them on the thumb and right index of his hand. A wide visualization screen was placed in front of the screen and a desktop, where during the experiment the subject was invited to place its elbow. A sequence of 27 objects was presented twice to each subject, for a total of 54 runs performed by each subject. All the objects in the randomized sequence were cubes with the same geometry, with pseudorandom changes in the weight m (0.1,0.2,0.4 Kg), in the friction coefficient, both static μ_s (0.4, 0.8, 1.2) and dynamic μ_d (0.3, 0.6, 1.1), and in the stiffness k(0.5,1, 2 N/mm). In each randomized sequence all the possible combinations of weight, friction and stiffness, without repetition, were presented to the subject. The values of μ_d were univocally associated to μ_s. The experiments were conducted with one grasping only condition, with the object hold between index and thumb tip of the same hand. Values for friction coefficients were assumed by [79] where experimental values of linear friction are reported between index tip and different materials, equal respectively to 0.42, 0.61 and 1.67 for rayon, suede and sandpaper.

Methods

The experiment consisted of a series of test runs. At the start of the experiment, one object with the shape of a cube was visualized at the center of scene.

Each subject was asked to grasp the object by index and thumb fingers, and to get acquainted with the weight of the object, by lifting it up and letting it falling down by continuously decreasing the gripping force.

After the necessary time to get acquainted with the object, the subject was asked to hold the object stationery in the air for 10 seconds with the minimum grasp force that he considered necessary, with the elbow leaning on the plane. When the subject was holding the object in the fixed position, both grasp F_n and friction F_t forces (respectively normal and tangential to the object surface) and positions of finger tips, object and proxies were recorded for each contact point. Statistical analysis was performed in SPSS 13.0.

Results

A significant correlation was found between the values of the gripping force F_n and stiffness, weight and friction values, as shown in table 2.2. The value of grip force F_n was found to be significantly positively correlated with mass and stiffness, while negatively with friction value. Table 2.2 reports the correlation coefficients obtained with a Spearmann non parametric test and significance level $p < 0.001$.

Table 2.2: Correlation table ($^{**}p < 0.001$).

Correlation measure	
	Grip force F_g
Friction μ_s	-0.346^{**}
Mass m	0.391^{**}
Stiffness	0.333^{**}

Figure 2.4 presents several bar plots comparing the grasping force F_g in different conditions according to change in friction (top-bottom) and mass(left-right). Different colors are used for clustered bars representing the effect of stiffness in each test run.

In Figure 2.5 it is shown the change of grip force F_g vs. weight, with superimposed the error bars with confidence interval of 95%, for 9 different conditions given by different combination of friction (bottom-top) and stiffness (left-right).

Discussion

From the analysis of the results, it can be seen as greater gripper forces are required for holding heavier weights and stiffer objects, while lower gripper forces are required for higher friction values. This confirms the empirical laws that have been already found in the case of manipulation of real objects with bare fingers. In [47] it was found that the relative safety margin, defined as the safety margin in percent of the grip force was about constant during lifting with increase of weights, was almost constant with change of weight. The calculation of the safety margin in the case of virtual manipulation allows to make an interesting comparison. As it is shown in the logarithmical plot in Figure 2.6 in the case of virtual manipulation, the safety margin tends to be reduced with increasing weight of the lifted mass.

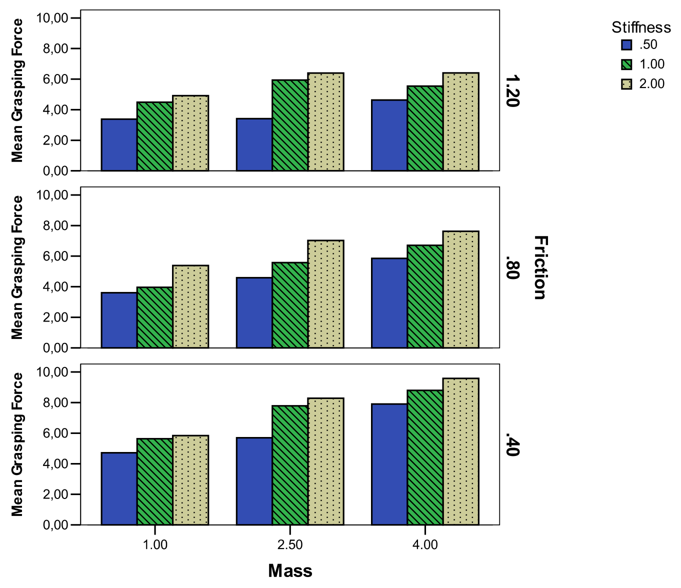


Figure 2.4: Mean grip force as a function of stiffness, friction and mass values

This was due to the larger dispersion of grip forces observed for lower mass values. In fact, due to the absence of local sensation of slip, it was more difficult to discriminate the weight of lighter objects. Moreover lighter object required a smaller resolution in the control of force (ΔF), that is limited by the position resolution of the device ΔX, according to the law $\Delta F = k\Delta X$, where k is the simulated contact stiffness. This is confirmed by the finding that better safety margins are obtained for lower values of the stiffness, as it is evident from the plot of Figure 2.7, where grip forces are plot vs. slip forces. While the minimum required grip force is represented by the diagonal line, experimental data can be divided according to the value of contact stiffness during the simulation.

2.3 Soft Finger Contact Model

In the following we will present two algorithms that can be used to simulate the haptic interaction between a set of human fingertips and a virtual object. The first algorithm proposed is simpler to understand and can be easily added on top of pre-existing state of the art haptic rendering algorithms supporting point-contact interaction. The second algorithm is based on more complex mathematical foundations, making it more difficult and harder to implement on top of pre-existing algorithms, but is capable of simulating the interaction between linear and rotational friction effects of the human fingertips in a more realistic way. It is important to note that both algorithms are independent of the type of model chosen to simulate rotational friction between a human fingertip and an object. For a review of possible models see [10].

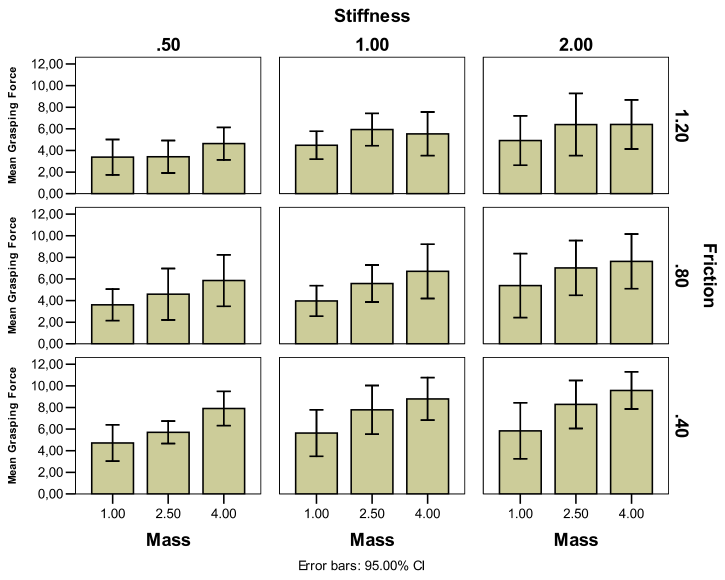


Figure 2.5: Mean grip force and as a function of stiffness, friction and mass values

2.3.1 The proxy algorithm with uncoupled friction

A 4 DOF god-object can be used to simulate a soft finger contact. Three of such degrees of freedom describe the position that the point of contact would ideally have when touching a virtual object, as for the standard god-object algorithm with linear friction [9]. A fourth variable is added to describe the relative angular motion between the two soft finger avatars and a virtual object. It is important to note that the two parts of the algorithm are disconnected, i.e. they do not influence each other in any way.

When a soft finger avatar comes into contact with a virtual object α_p is set to the current value of the angle describing the rotation of the soft finger avatar α_0. The position of the haptic interface is described by the position of the HI point $\mathbf{x_h}$. The following steps are then performed until contact is not broken.

At a generic k-th time sample:

a: Computation of goal position. The new goal position for the god-object is computed as $\mathbf{x_g} = \mathbf{x_s}$, where $\mathbf{x_s}$ is the surface point which minimizes the distance between the HI point $\mathbf{x_h}$ and the contact surface. The new angular position of the userŠs fingers is calculated as $\alpha_g = \alpha_s - \alpha_0$, where α_s is the angular rotation measured by the haptic device.

$\mathbf{x_g}$ and α_g are assumed as the new goal values respectively for $\mathbf{x_p}$ and α.

We assume the following definitions:

$$\begin{cases} r = & \|\mathbf{x_g}(k) - \mathbf{x_p}(k-1)\| \\ \rho = & |\alpha_g(k) - \alpha_p(k-1)| \\ d = & \|\mathbf{x_g}(k) - \mathbf{x_h}(k)\| \end{cases} \tag{2.25}$$

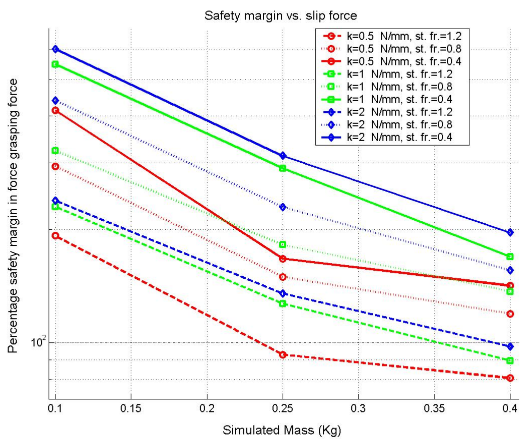


Figure 2.6: Relative safety margin during grasping vs. manipulated mass for different condition

b: Analysis of the friction condition. In static conditions the new position of the god-object can be expressed as:

$$\begin{cases} \mathbf{x_p}(k) = \mathbf{x_p}(k-1) & \text{if } \frac{|F_t(k)|}{\mu_s|P(k)|} = \frac{r}{\mu_s d} < 1 \\ \alpha_p(k) = \alpha_p(k-1) & \text{if } \frac{|M(k)|}{\Gamma(P(k))} = \frac{k_r \rho}{\Gamma(P(k))} < 1 \end{cases} \tag{2.26}$$

where $|P| = k_l d$ is the force directed along the contact normal and $\Gamma(P)$ depends on the model chosen for the rotational friction and k_l and k_r are the haptic servo-loop gains, equivalent to a linear and rotational stiffness, used for calculating the elastic penetration force and torque.

If a linear approximation is used for the function $\Gamma(P(k)) = \mu_r P(k)$, then the second condition can be rewritten as:

$$\frac{k_r \rho}{k_l \mu_r d} < 1 \tag{2.27}$$

Otherwise, conditions of dynamic friction should be applied and the god-object, sliding over the surface, is moved on the boundary of the dynamic friction cone:

$$\begin{cases} \mathbf{x_p}(k) & = \mathbf{x_g}(\mathbf{k}) + \mathbf{r}' \\ \alpha_p(k) & = \alpha_g(k) + \rho' \end{cases} \tag{2.28}$$

with

$$\begin{cases} \mathbf{r}' & = \frac{\mathbf{x_p}(\mathbf{k-1}) - \mathbf{x_g}(\mathbf{k})}{r} \mu_d d(k) \\ \rho' & = \frac{\alpha_p(k-1) - \alpha_g(k)}{\rho} \frac{\Gamma(P(k))}{k_r} \end{cases} \tag{2.29}$$

In case of a linear approximation for the Γ function, the equivalent condition is reduced to:

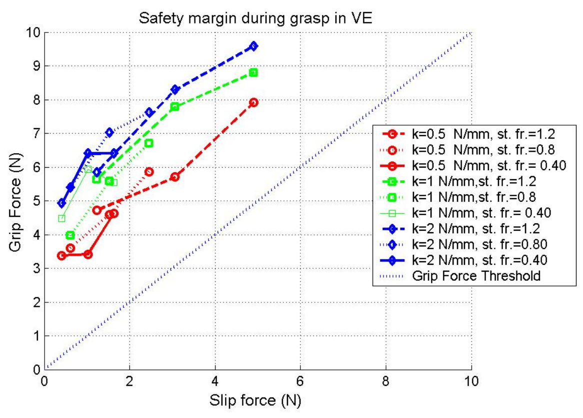


Figure 2.7: Grip vs. slip force, showing the relative safety margin during grasping of virtual objects

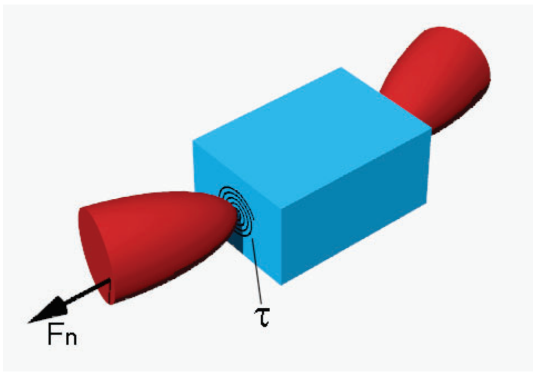


Figure 2.8: Soft Finger Proxy for the grasping of virtual objects

$$\rho'(k) = \frac{\alpha_p(k-1) - \alpha_g(k)}{\rho} \frac{k_l}{k_r} \mu_r d(k) \tag{2.30}$$

c: Computation of friction force and torque. A new torque $M(k) = k_r(\alpha_p(k) - \alpha_g(k))$ and a new force $\mathbf{F}(k) = k_l(\mathbf{x_p}(k) - \mathbf{x_g}(k))$ are computed using the new value of α_p and $\mathbf{x_p}$. Torque $-M(k)\mathbf{v_n}$ and force $-\mathbf{F}(k)$ are applied to the virtual object (where $\mathbf{v_n}$ represents a unit vector with direction along the contact normal). A force $\mathbf{F}(k)$ and a torque $M(k)\,\mathbf{v_n}$ are also applied to the user (if the device used is capable of actuating such wrench).

d: Computation of the new position of the object. New velocity $(\mathbf{v}, \boldsymbol{\omega})$ and position $(\mathbf{x}, \boldsymbol{\theta})$ is computed for the virtual object. Angle α_c representing how much the object has rotated about axis $\mathbf{v_n}$ is computed as

$$\alpha_c = |\boldsymbol{\omega} \cdot \mathbf{v_n}|\,\Delta_T \tag{2.31}$$

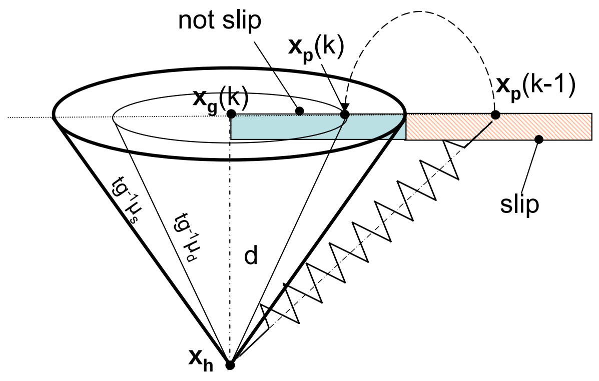


Figure 2.9: The classical friction cone for simulation of linear friction

where Δ_T is the servo-loop sampling time.

e: Update of god-object position. The current value of α_p and $\mathbf{x_p}$ are corrected to take into account the displacement of the virtual object:

$$\begin{cases} \mathbf{x_p} = \mathbf{x_p} + \mathbf{x_c} \\ \alpha_p = \alpha_p + \alpha_c \end{cases} \tag{2.32}$$

and then repeat from step a.

The dynamic behavior of this algorithm can be analyzed observing the plane described by the two axis, r and ρ where r is the absolute distance proxy-god while ρ is the absolute distance between the rotational proxy and the roational god. For every penetration depth d we identify two point over each axis corresponding to the two radii of the friction cones, where the linear is μd and the rotational si $\mu_r d$. The static friction condition for each component is identified by a value less than such radius, while in the case of dynamic friction a point above the limit is moved toward the radius. When considering now the two dimensional coordinates (r, ρ) the algorithm keeps the point in its position if it is inside the rectangle, while it moves the point to the nearest point on the rectangle if it is outside. Such behavior is shown in Figure 2.10.

Eventually we can normalize such graph respect the penetration depth for better understanding the dynamic behavior during the sliding of the contact points. If the object is sliding the proxy moves horizontally from an external position to $r = 1$ and eventually inside if it enters the static mode. Correspondly during the rotation the movement is vertical to $\rho = 1$. Finally in the case of both sliding and rotation the point lies on $(1, 1)$.

2.3.2 Analysis of the grasping conditions

If we take the simplified case of an object with a box shape held between the finger we can identify a series of curves for the sliding of the object depending on the applied pressure P and

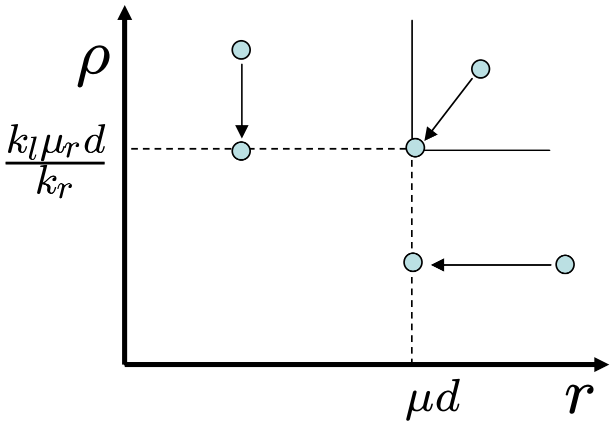


Figure 2.10: The plane r, ρ under the not coupled algorithm

the length L of the torque applied by the weight force over the rotation axis passing by the two contact points.

The condition for the non sliding state is:

$$\mu P \geq mg \tag{2.33}$$

While for the rotational component given that the istantaneous rotation axis has a distance L from the center of mass:

$$M \geq mgL \tag{2.34}$$

From the previous articles we know the relation between M and the pression P:

$$M/P = 3/16\pi\mu a(P) = k_r \mu a \tag{2.35}$$

Alternatively:

$$M/P = \frac{1}{2}\mu a \tag{2.36}$$

In the first case we define $k_r = 3/16\pi \ = 0.5890$ while in the second is $k_r = 0.5$. We define μ_r as:

$$\mu_r = k_r \mu a \tag{2.37}$$

With the chosen μ we have $\mu_r = 0.0787m$. From the research by Johansonn the experimental value of the μ_r for the fingerpad is $\mu_r = 0.01011m$.

The relation above becomes:

$$k_r \mu a P \geq mgL \tag{2.38}$$

$$P \geq mg\frac{L}{\mu_r} \tag{2.39}$$

Finally:

$$P \geq \frac{mgL}{\mu_r} \tag{2.40}$$
$$P \geq \frac{mg}{\mu} \tag{2.41}$$

When the two condition for sliding are threated separately for a pressure P that goes from infinity to zero we have two different behaviors depending on the length L. The object starts to slide if L is less than the L_0 on which the two conditions are equal, instead for L is greater than L_0 the object first starts to rotate then to slide and rotate.

$$\frac{mgL_0}{\mu k_r a} = \frac{mg}{\mu} \tag{2.42}$$

Using the theorical values we obtain $L_0 = 4.8cm$ that independent from the friction coefficient.

When the rotational friction is expressed explicitly by μ_r in this case the above equation should be changed:

$$\frac{mgL_0}{\mu_r} = \frac{mg}{\mu} \tag{2.43}$$

$$L_0 = \frac{\mu_r}{\mu} \tag{2.44}$$

From the above and the experimental values we obtain that $L_0 = 6mm$, a value that makes almost impossible to slide an object before rotating it.

The figure Figure 2.11 show the above two conditions for the linear and rotational sliding, when taking the experimental parameters.

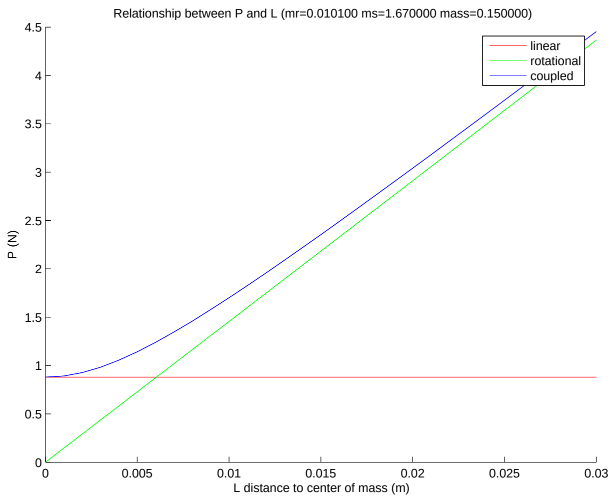


Figure 2.11: Condition between sliding and rotational. With the experimental mr and ms

The above sliding condition can placed now in the plane r, ρ for understanding the limits of the proxy positions r, ρ, as shown in Figure 2.12 assuming that $\Gamma(P) = \mu_r P$:

$$k_l r \geq mg/2 \tag{2.45}$$

$$k_r \rho \geq mgL/2 \tag{2.46}$$

Given the above conditions in the plane r, ρ it is easy to understand that there is a limit in the arm length L at which the rotational friction is not more able to prevent the rotation:

$$L_{max} = \frac{2k_l \mu_r}{mg} d \tag{2.47}$$

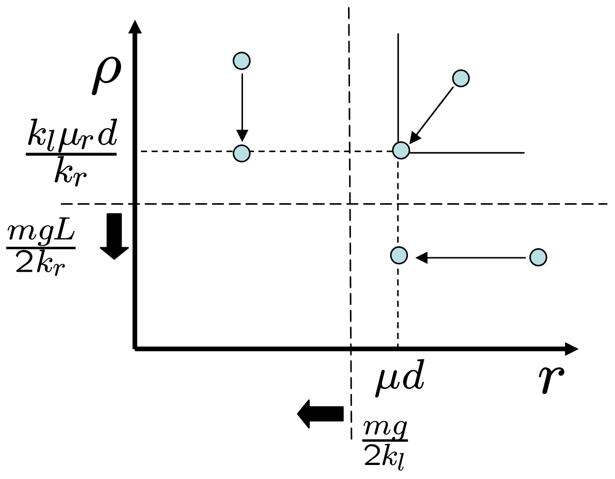


Figure 2.12: Plane r,ρ with the sliding and rotation conditions

2.3.3 Coupled Soft Finger Proxy Algorithm

Given the elements exposed above it is now possible to formulate a god-object algorithm that combines linear and rotational friction effects. At a generic k-th time sample, step **b** of previous algorithm is changed to **b'** as follows:

b': Analysis of the friction condition. Compute $\varepsilon = x^2 + y^2$ by:

$$\begin{cases} x = \frac{r}{\mu d} \\ y = \frac{k_r \rho}{\Gamma(P(k))} = \frac{k_r \rho}{\mu_r k_l d} \end{cases} \tag{2.48}$$

When the god-object is inside the equivalent friction cone, the position of god-object is not changed and so:

$$\begin{cases} \mathbf{x_p}(k) = \mathbf{x_p}(k-1) & \text{if} \quad \varepsilon \leq 1 \\ \alpha_p(k) = \alpha_p(k-1) & \text{if} \quad \varepsilon \leq 1 \end{cases} \tag{2.49}$$

If $\varepsilon > 1$, the god-object is sliding and the point $[r, \rho]$ is moved to $[r', \rho']$ on the the boundary of the corresponding $p-$curve, as it is shown in Figure 2.13. So we have:

$$\begin{cases} \mathbf{x_p}(k) & = \mathbf{x_g}(\mathbf{k}) + \frac{r'}{r}\left(\mathbf{x_p}(\mathbf{k-1}) - \mathbf{x_g}(\mathbf{k})\right) \\ \alpha_p(k) & = \alpha_g(k) + \frac{\rho'}{\rho}\left(\alpha_p(k-1) - \alpha_g(k)\right) \end{cases} \tag{2.50}$$

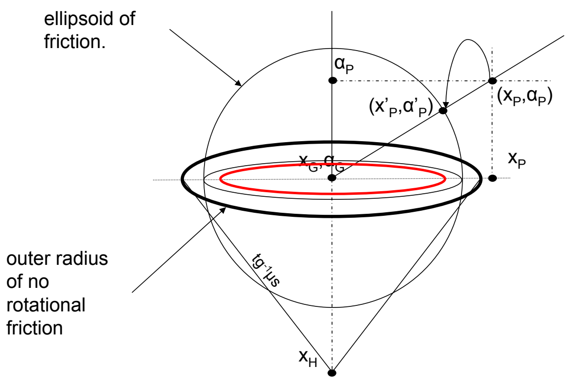


Figure 2.13: The interpretation of the mixed rotational-linear friction adaptive cone

The main question is now if there is any difference between the coupled and not coupled algorithms, and also if there is any benefit from introducing the coupled algorithm. By looking at the sliding conditions in Figure 2.14 it is possible to understand that the coupled algorithm allows to have an object that both slides and rotates around the fingers of the user. Actually the effectiveness of such algorithm depends on the crossing point L_0, because with small values of L_0 the two curves are similar with the effect of reducing the difference in behavior.

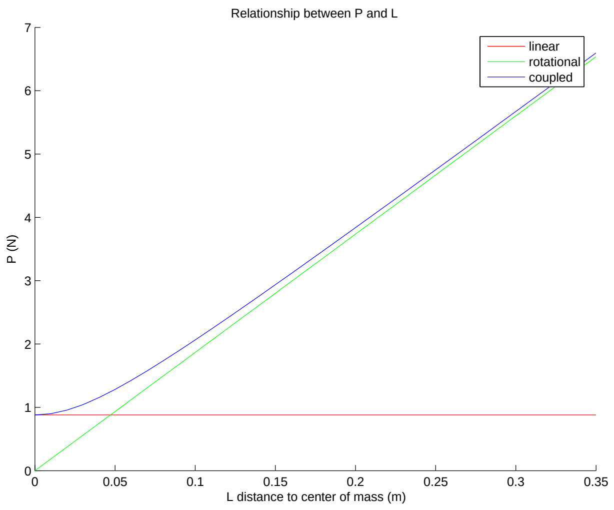


Figure 2.14: Condition between sliding and rotational. With the therical mr and ms

The rectangular condition in the plane r, ρ shown in Figure 2.15 is now changed into an area shaped as an ellipse. When the point r, ρ is outside the ellipse the proxy is placed on the nearest point. The Figure 2.15 shows the two conition regions and the behavior in the case of dynamic friction. First we need to observe that the condition for the static friction is different and the coupled algorithm triggers the dynamic friction earlier. Second during the dynamic friction status we have the contribution of both sliding and rotation.

If we use the friction coefficientd $1.67, 0.01011$ on this plane and measure r in meters we obtain an ellipse that is extremely stretched, that means that for small values of r the behavior of the algorithm is similar to the uncoupled independently by the rotation ρ.

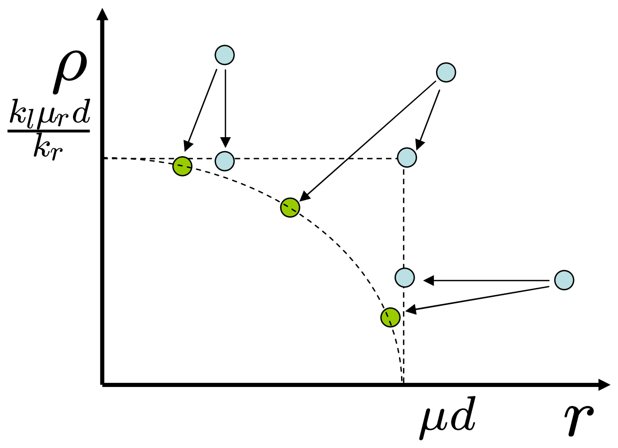


Figure 2.15: The plane r, ρ under the coupled algorithm

The conditions for sliding and rotation expressed in the plane r, ρ now introduce two possible behaviors depending on the parameters. In Figure 2.16 the left case shows that it is impossible to have a static behavior because every point on the curve is outside one of the two limits. The right case, instead, shows the region in blue in which the behavior is completely static. The second behavior is guaranteed if the limit point $\frac{mg}{2k_l}, \frac{mgL}{2k_r}$ is inside the ellipse.

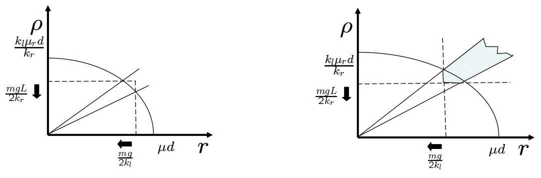


Figure 2.16: The coupled region with the condition that prevents static behavior on the left, and the static region on the right

Figure 2.17 shows the plane r, ρ during a simulation of the grasping with the coupled algorithm. First the obejct rotates and then it slides. Using the not coupled algorithm the same simulation does not allow to rotate the obejct.

2.3.4 Experimental validation and applications

The algorithm proposed above has been used in conjunction with a GRAB haptic device [5] allowing two-points interaction with virtual object (see Figure 2.18). The current design of the device does not allow to recreate contact torques on the users' fingertips. In this scenario the soft-finger algorithm is used solely to compute the effect of the user on the virtual environment. Work is currently being carried out in order to add rotational feedback on the user's fingertips.

The application consisted in the manipulation of a rectangular block, with its center of mass

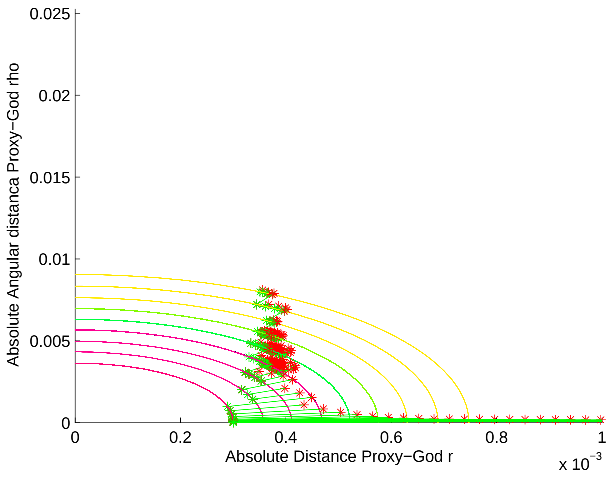


Figure 2.17: Rotation and Slide using Coupled

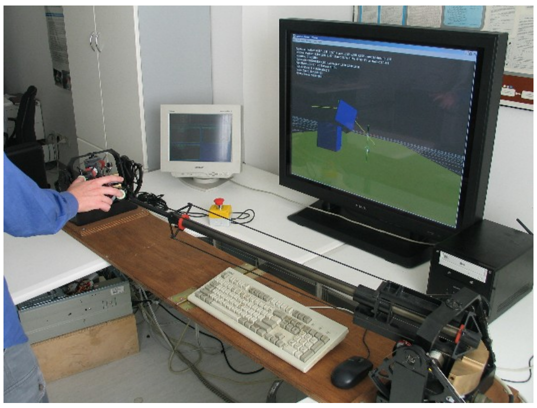


Figure 2.18: Manipulating virtual objects using two fingers per hand

not coincident with the gripping point GP, so that the gravity force exerted a moment with respect to GP. The movements of the block were constrained to the vertical plane, so that only displacements and rotations in this plane were allowed.

The subject was asked to grasp the block on the two opposite planar faces with his two fingers of the same hand, and then to slowly release the pressure between the fingers until the object started sliding. The starting position of the block was horizontal. The subject was instructed to modulate the gripping force in order to achieve a rotation of the object between the fingers, with no or limited sliding. He was allowed to regrip to get out of the sliding state and back into a holding state, when he was not able to achieve a correct rotation of the block.

Different configurations were tested: in particular a class of rayon-suede like materials and sandpaper-like were simulated using the values reported in table 2.3. Two reference conditions are herewith reported, under the hypothesis of a sandpaper-like material with $\mu_l = 1.67$ and $\mu_r = 10.11$ mm:

Case A: coupled linear and rotational friction;

Surface material	μ_l	μ_r mm	μ_r/μ_l mm
Rayon	0.42 ± 0.07	3.05 ± 0.57	7.39 ± 0.91
Suede	0.61 ± 0.1	3.84 ± 0.74	6.37 ± 0.88
Sandpaper	1.67 ± 0.24	10.11 ± 1.50	6.20 ± 1.54

Table 2.3: Experimental values of means $\pm SD$ of μ_l, μ_r and their ratio measured at the index fingertip for different materials

Case B: uncoupled linear and rotational friction.

Figure 2.19 represents a typical motion on an object grasped between the fingers, when the grip force is slowly released in condition A. It is easy to see from figures Figure 2.19 and Figure 2.20 that there is a translational component of the movement associated to the rotation. Modulating the exerted pressure the subject was able to achieve a smooth rotation of the block with a small amount of translation, as it is visible from the analysis of angle α vs. time, plotted in Figure 2.20.

The repositioning of the god-object on the $p-$curves is shown in Figure 2.21. The sliding over the p-curve represents a movement with constant pressure and a reconfiguration of the instaneous center of rotation. The position of the god-object before repositioning, in dynamic friction conditions, is shown by the diamond black markers: the god-object is then moved on the corresponding $p-$curve, through the connecting line shown in the same figure. The increase of pressure allows to block the rotation of the object, that is represented by the reaching of the outer $p-$curve in Figure 2.21, that can provide an adequate value of friction forces to stop the rotation of the object.

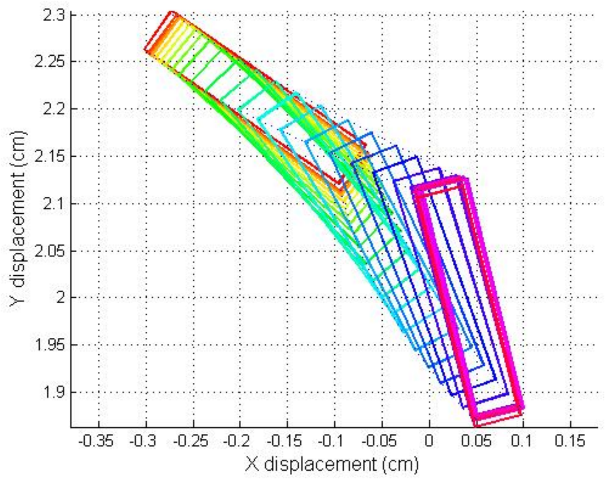


Figure 2.19: The movement of a rectangle block grasped among two fingers under condition A

In condition B the subject was not able to let the object rotate without sliding between his fingers. From the analysis of Figure 2.23, it can be seen how the rotation associated to the sliding

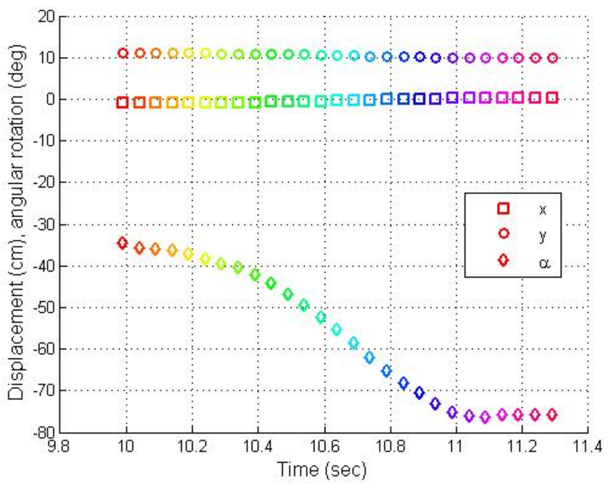


Figure 2.20: Trajectory vs. time (x and y of CM and rotation α around GP) of the simulated motion under condition A

movement is lower than in condition A and not as much smooth.

The risultant motion is shown in Figure 2.22. The rotation cannot be controlled by the subject, who is holding the object modulating the grip force; at the end the object falls down without changing its initial orientation when the grip force is gradually released by the subject.

These performed tests revealed that using experimental physiological parameters for the friction coefficient, the two algorithm behave in a very similar way, because of the small value of the L_0 parameter. The new proposed friction algorithm allows a more realistic simulation of sliding and rotation togheter only when this value is in the oder of 10mm.

In a second experiment we asked the users to release the object and make it align to a reference object that was oriented with a specific angle (30 or 60 degrees). Each person is presented with the object having different lengths, masses and the coupling enabled or disabled. This experiment is represented schematically by the Figure 2.24, while the result for the users is presented in Figure 2.25. The alignment error is smaller for the coupled algorithm although some issues are still present for a very small values of L caused by instabilities in the dynamic simulation.

2.4 Discussion

The grasping models discussed above allow an efficient and effective way to simulate the grasping in haptic environments. We have presented both the solution with sliding and another more complete solution in which the rotational friction has been taken into account. The main limitations in the above solutions are introduced by the haptic interface itself because the dual arm 3-DOF device used is not able to provide not only the tactile feedback necessary to feel the sliding but also the 6-DOF feedback that is caused by the rotational friction.

A possible approach that could overcome the above limitation is a sliding feedback force sent to the user when the object starts to slide or rotate. This force is a haptic hint that if correctly

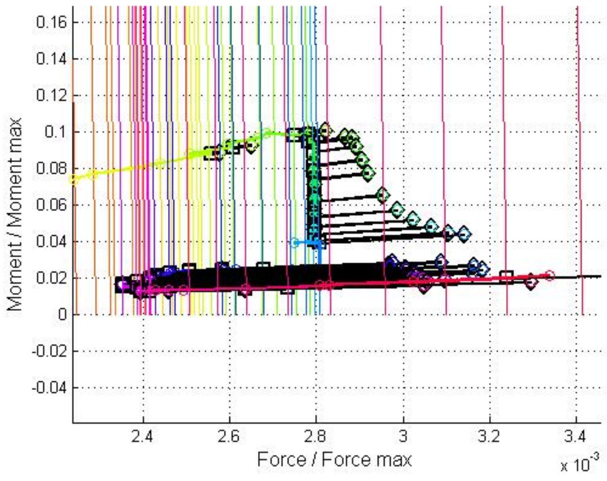


Figure 2.21: Representation of the repositioning of the god-object in the $[r, \rho]$ plane under condition A

interpreted by the user would be more effective than seeing the graphical representation of the object sliding between the contact points. A good candidate for such force effect would be a vibration along the vertical axis in which frequency and amplitude should be proportional to the sliding of the body. Clearly such a feedback is not natural as the tangential strain caused by the real sliding but it is a hint that can be perceived at much higher rate than the visual one.

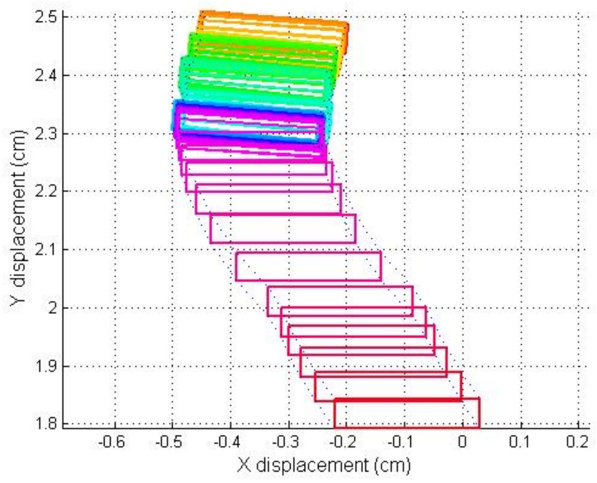


Figure 2.22: The movement of a rectangle block grasped among two fingers under condition B

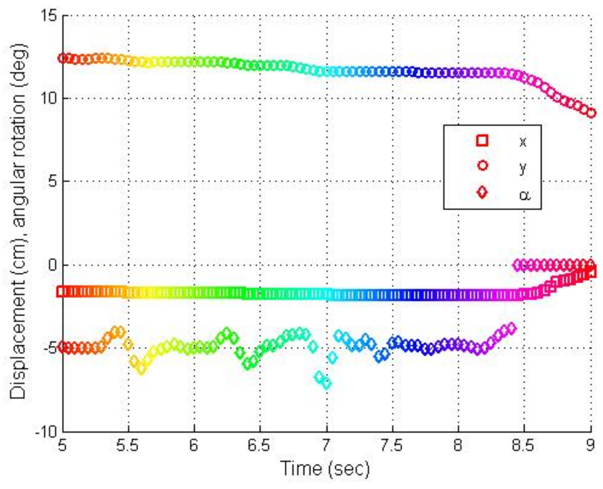


Figure 2.23: Trajectory vs. time (x and y of CM and rotation α around GP) of the simulated motion under condition B

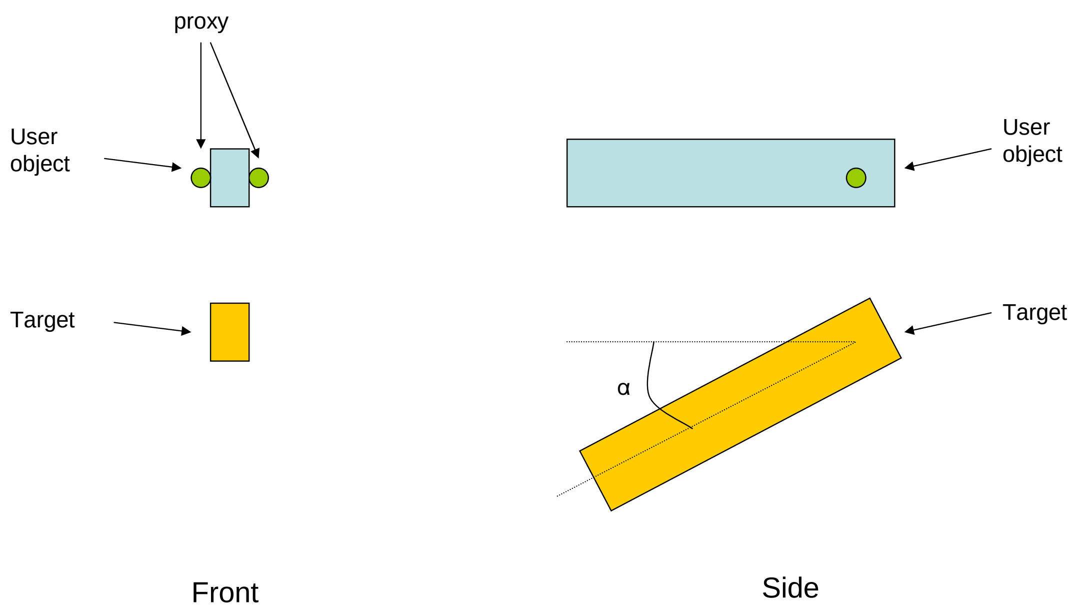


Figure 2.24: Schematic representation of the experiment

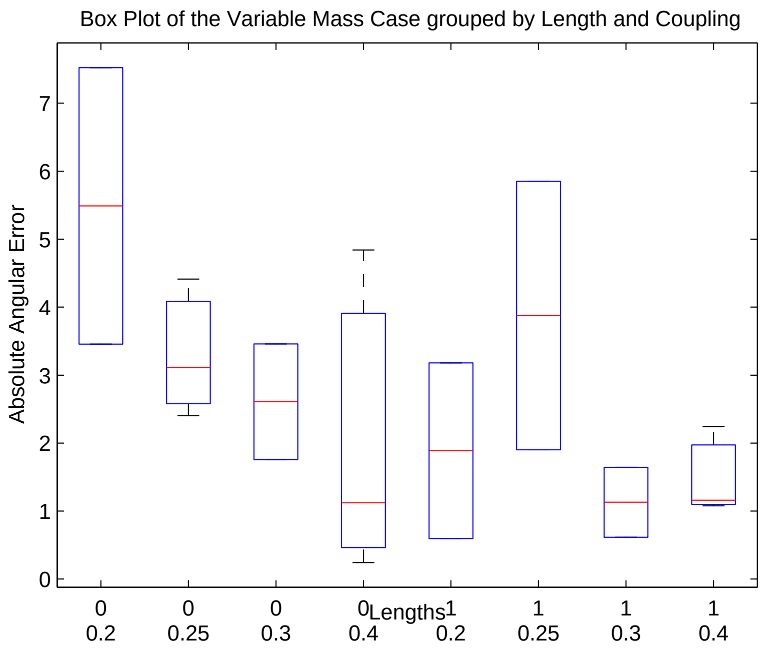


Figure 2.25: Box Plot of the alignment error with different lengths and the two algorithms

Chapter 3

Voxel Based 6DOF Haptic Rendering

This chapter presents a novel Collision Detection and Response algorithm for the 6DOF haptic rendering of objects based on Voxel Volumes. This algorithm is based on an implicit sphere tree representation that allows the efficient storage and test of the collision. The implementation of this algorithm has been applied in a tool for the planning of craniofacial surgical operations, for the manual alignment of models obtained from CT scans.

3.1 Introduction

The interaction between physical bodies is something that happens in every moment of our life. Bodies collide between each other and they interact by moving away. In this scenario the stiffness of the bodies, their geometry, their mass and friction play a fundamental role in the determination of the resulting behavior. A more interesting case is constituted by the human manipulation of one of the two bodies for the achievement of a task. The task in question could be the insertion of one of the bodes inside one hole of the other or finding a configuration of the two bodies in which the surfaces are matching.

This work has been developed in the context of a project with the aim of providing a haptic interaction technique based on a voxel volume representation of the bodies able to provide realistic six degrees of freedom feedback. This result is achieved by a new collision detection algorithm between voxel models and a collision response algorithm for computing the resulting force feedback.

An haptic interface allows to simulate the interaction with a real object by responding with a force feedback to the movement of the user's body inside the physical space. The movement in the physical space is transferred into a movement inside the virtual environment and the force feedback is computed by considering the interaction of a virtual body with the other bodies inside the environment.

In the simplest case the virtual body is represented by a small sphere, usually considered as a point, and the force feedback is computed as a directed force, without taking into account the rotational components. The proxy based algorithms provide this kind of interaction and are useful when simulating the exploration of surfaces or the perception of force fields [122, 148]. The simulation of haptic interaction between objects has been addressed by 6DOF algorithms that involve a more complex collision detection and a dynamic simulation of both the body and the virtual body.

3.1.1 Collision Detection

Collision Detection is a branch of computer science and robotics whose objective is to identify the collisions between multiple moving objects. Many different CD algorithms exist and they differ in the geometrical representation of the objects, the precision of the result and their computational requirements. The selection of a CD algorithm depends also on the specific application like the case of haptic interaction, cloth simulation or game environment. In the case of 6DOF haptic interaction the focus is both on responsiveness and on quality of rigid body collisions.

A collision detection algorithm takes the input geometry expressed usually in the form of triangular meshes and constructa additional data structures for improving the collision computation. Some algorithms pose constraints on the type of geometry used like convexity or absence of holes. When multiple bodies are involved the first step of CD consist in the identification of which bodies are possibily colliding, an operation that for haptic interaction is reduced to the identification which environment body collides with the haptic probe. The effective collision detection is usually performed by evaluating a hierarchy of bounding volumes that allow to perform the collision detection at various levels of detail. At the lowest levels of the hierarchy the collision computation is explictly performed by computing the collision between the geometrical elements. In some cases the collision detection algorithm is a collision avoidance algorithm in which the possible collision is triggered at a certain distance before the effective interpenetration of the objects.

The result of the collision detection algorithm is a set of contact points each described by the following information: the contact point, a penetration depth and a contact normal. The surface of the contact is not generally taken into account and only in [107] is used for limiting the descent in the collision tree.

The above description is valid for the case of static collision detection in which time is not involved. The presence of time has two effects on the collision detection algorithms: the first is that is necessary to take into account the velocity of the bodies and consider if inside a specified time step the moving bodies can collide. [20],[65]. The second is the possibility of using time coherency to reduce the collision computation time. Instead of searching for contact at every iteration it can start from the last contact points performing a local search for the new contact.

A contact can be described by a pair of points $\mathbf{p}$ and $\mathbf{p}_0$ over each object interacting, a normal $\mathbf{n}$ that we assume going outside the grasped obejct and a distance d that is positive if the object penetrate or negative if they are not penetrating but inside the threshold distance t. Note that in most algorithms a contact with a separating contact points velocity is discarded for the computation of the response.

The management of contact points is important for providing a stable interaction because a varying number of contact point induces instabilities in the collision response and it is also more important when dealing with resting contacts [78],[144]. When dealing with multiple collision points between objects we can have *multiple contact regions* and in each region *multiple interference points*. Clustering is an operation that can be performed to reduce the number of contact points and corresponds to reduce them to the single contact regions. A cluster groups point inside a certain threshold distance [78],[77] or group them to form at most K clusters. In the case of resting contacts is necessary to identify the movement of the contacts for their identification as resting. In this case the contact velocity and a thresholding distance are used to identify existing contacts respect new contacts [34]. For every cluster a representative contact is computed by taking means of the contained contact points. Example of possible means are the following.

$$\mathbf{n} = \frac{\sum(t-d_i)\mathbf{n}_i}{||\sum(t-d_i)\mathbf{n}_i||} \tag{3.1}$$

$$\hat{\mathbf{p}} = \frac{\sum(t-d_i)\hat{\mathbf{p}}_i}{\sum(t-d_i)} \tag{3.2}$$

$$d = t - max(t-d_i) \tag{3.3}$$

The clustering operation with threshold δ can be obtained by using an octree that stores leaf cells with diagonal δ, or alternatively by a k-mean approach that finds at most K clusters from a given set of points [1],[72].

Distance Algorithms

At the finest level of the collision detection hierarchy there are the geometric distance algorithms, that taken two geometric primitives compute the nearest points. A selection critaeria for a collision algorithm should take in to account both the computational time, the robusteness respect numerical error and the quality of the implementation. In the case of convex polyhedra the reference algorithms are GJK [49], [21], [142], Lin-Canny [85], [29]. V-Clip [97], and others [7],[41],[58],[43]. With these algorithms the nearest points between two convex polyhedra are computed in almost constant time given a starting search point.

A totally different approach has been adopted by Johnson . In this case the algorithm is using collision avoidance and it computes the local minimum distance between to general polyhedra.

Penetration Depth

The distance between two convex polyhedra is not sufficient for the collision detection computation. We need to compute the penetration depth of the bodies, defined as the minimum distance that is necessary to move one of the two along the contact normal to avoid the collision. In the case of convex polyhedra is possible to use the internal Voronoi region to compute the penetration depth [112], while [102] ([61]) is the original algorithm on the topic.

Bounding Volumes Hierarchies

Instead of performing the direct test between primitives or convex meshes a hierarchical bounding volume structure allows to reduce the collision time by avoing the collision tests among the parts of geometry that are too distant. A bounding volume is a simple geometrical primitive that is able to contain one or more leaf geometries and is the structured in a hierarchy of bigger bounding volumes. The hierarchy of the primitives is usually a binary tree, that is constructed balancing the distribution of the nodes.

The type of primitive used for the bounding volume is the characterizing element of each algorithm and many primitives has been evaluated each with different tradeoffs. The most famous have been Spheres [66], Axis Aligned Bounding Boxes [141],[146], Object Bounding Boxes [53] and k-DOPS [81]. Each primitive has different costs:

- Storage cost
- Construction cost - for producing an optimal tree
- Cost of the collision test between primitives
- Update cost after an affine transformation

Among the different solutions the a sphere tree has been extensively used for deformable objects because it can be easily updated, although it has the draw back of occupying more space than other primitives.

A particular case of Hierarchical Bounding Volume is the multiresolution approach presented by [107] in which the leaves are convex decompositions of the original object and the intermediate nodes are convex representation at lower resolution of the children nodes.

Space Partitioning Techniques

While the Hierarchical Bounding Volumes decompose an object in a hierarchy of volumes, the space partitioning techniques decompose the space in smaller spaces for the same objective of collision detection. The difference among the two approaches is in the type of query that is expected to be performed.

Binary Space Partioning algorithms are the classic example of Space Partitioning in which the space is divided into two sub spaces depending on a cutting plane.

A different technique is adopted by Octree structures in which the space is recursively divided into cubic sub spaces till the object is decomposed with all the details.

Collision Detection in Haptics

The collision detection for haptics can be performed at the full haptic rate only when it guarantees a constant execution time like in [144] otherwise most approaches require a slower collision detection task that is sided by an incremental algorithm executed at haptic rates. In most the cases a BVH based on GJK or Lin-Canny is used for polygonal meshes. [2], [107], [77], [96]

3.2 Voxel Collision Detection

This section presents the voxel collision detection algorithm and the associated storage model. First we present the general overview of the algorithm and its placement in the organization of Volume algorithms, then follows a description of the implicit representation, some optimizations and finally the latests aspects about sensation preserving for improving the performance.

3.2.1 Overview

The Collision Detection module of this algorithm is based on the idea of using a Bounding Volume Hierarchy of Spheres that are implicitly built from the Hierarchical representation of an Octree. The rationale for this is to cope with the memory occupation requirements of Volume Models while at the same time use Spheres for testing of the collisions. As discussed before spheres are extensively used because the low cost of test, update and their invariance respect the rotation of the bodies. For example AABB are efficient for ray testing applications but when two bodies in two different reference systems need to be compared they are converted into OBB.

Among the different volume modeling options we have selected the Octree, or the generalized spatial tree, because it is a natural hierarchical structure for volume models. The Volume Model itself is represented by surface type markers (Empty, Proximity, Surface and Full) that are useful for optimizing the collision. Optionally in some variants we have used distance fields.

The volumes used in this document has been obtained from triangular meshes using a marching voxelization algorithm or directly from an imaging device.

The Figure 3.1 shows the overall structure of the algorithm both in terms of Volume Modeling and in terms of 6DOF Haptics.

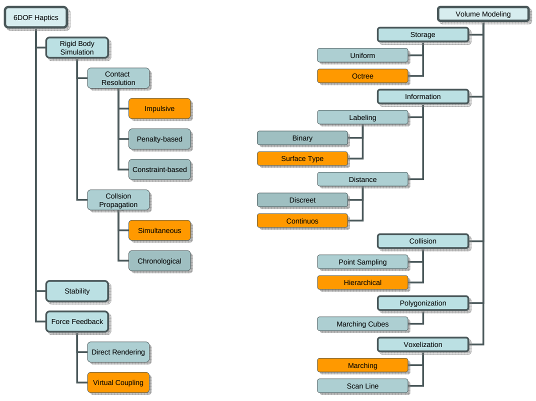


Figure 3.1: Selected characteristics for the new algorithm

3.2.2 Implicit Representation

The core of this algorithm is a bounding volume hierarchy based on spheres. In this paper we propose a modified octree structure that maintains a bounding sphere around each node with no additional storage and constant-time computation. For a cube of side x the sphere has a radius of $x\sqrt{3}/2$. When describing the octree structure we start from the voxel size s and build the octree of side $2s$. We describe the octree in terms of levels starting from the voxels that have a level of 0. The root of an octree containing 256 voxels per side has level 8. Given the level L we can compute the radius of the root as $s2^{L-1}\sqrt{3}$. In the case of the generalized N-tree the radius is $s2^{NL-1}\sqrt{3}$ where N is 1 for the octree, 2 for the 64-tree and 2 for the 512-tree.

Given a sphere at a certain level we are able to compute the spheres of every child by scaling the radius by 2 and offsetting the parent radius by $s2^{L-2}$ along the three directions.

3.2.3 Collision Detection

The Collision Detection is performed by starting from the root of the two objects and representing the two spheres in global coordinates. If the two root spheres are colliding we detail the collision by comparing each children sphere of the first with the second sphere. Every recursion of the collision tree we descend the level of only one object and then we test the children of the other. This operation is performed till the lowest level of both the objects are reached. This full test can be optimized by using surface only octrees, normal cones or sensation preserving approximation of the contact.

The descent from one sphere to the ones of the children needs to be performed in the world coordinates. Because of the fixed structure of the octree at the beginning of the Collision Detection iteration we precompute the translation vector of every one of the 8 children expressed in world coordinates and then when needed we apply the scaling factor depending on the level. This approach allow us to perform the local-world coordinates only in this small precomputation phase and work always in world coordinates.

Because we are interested in the contact between surfaces it is possible to simplify the collision detection by testing only the surface voxels. This simplification can be done during the octree construction phase or during the collision detection by taking the children bitmask and masking it with the surface bitmask.

3.2.4 Optimizations

The first optimization can be applied to the computation of the sphere hierarchy. When the octree (or the generalized n-tree) is not full we could compute a bounding sphere that takes into account the real distribution of the children of the octree node. This optimization has the objective of reducing the volume of the sphere and in general make it tighter reducing the number of collision tests.

The Figure 3.1 shows the case of the four possible cases of bounding circles in the case of a quadtree. In the quadtree case we have only 15 possible combinations of children that give 3 full sphere cases and other 12 cases with three circle types. When we move to the octree we have 255 combinations among which 85 are not full spheres. Because the layout of these spheres is size invariant we precompute it using the Ritter algorithm for obtaining the sphere that contains the vertices of the children of the octree. For each of the 255 cases we mark if the sphere is a full sphere otherwise we store the radius and the translation respect the center, to be scaled by the current radius of the octree sphere.

When handling the collision (A_i, B_j) we can skip one level of testing if one of the two nodes has a single child because the bounding sphere of the child corresponds to the bounding sphere used during the testing of the parent. The Figure 3.3 shows the skip case with a quadtree.

The recursive collision detection function *collide* receives two nodes (A_i, B_j) that are known to be colliding, and it tries to use the information about the level and number of children of the two nodes. For example when A contains only leaves, that is $i = 1$:

- $j = 0$ and $count(A_i) = 1$ then we have a contact between two voxels
- $j > 0$ and $count(A_i) = 1$ then we call $collide(B_j, A_0^k)$ for every voxel k in A

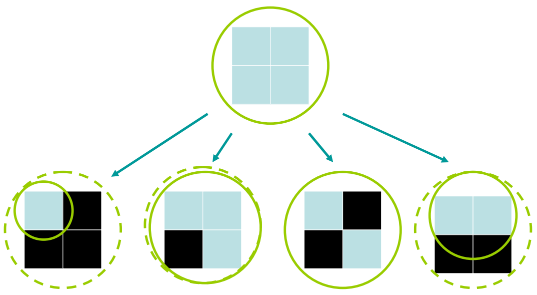


Figure 3.2: Example of sphere tighting depending on the number of children in the quadtree

- $j = 0$ and $count(A_i) > 1$ then we test the voxels directly
- $j > 0$ and $count(A_i) > 1$ then we test the voxels of A with B and eventually $collide(B_j, A_0)$

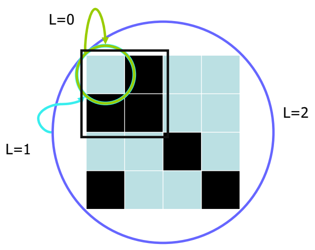


Figure 3.3: When an octree has a single child the collision detection can skip one level

An additional improvement that simplifies the collision detection between the two sphere hierarchies could be introduced by using the Normal Cone information as in [74]. Because of the spatial placement of the children octreees and their associated implicit spheres it could be easier to have separation of the cones. In this work we optionally store the Normal Cone associated with every octree node and use it to reduce the number of tests. Respect [74] here the normal cones are used for reducing the number of test and not for providing a global search of minima between the primitives. Later we show the performance improvements of using normal cones respect their additional memory requirement.

3.2.5 Sensation Preserving

The idea of Sensation Preserving comes from [107] in which the collision detection algorithm limit the descent in a multiresolution tree by evaluating the loss in user perceived sensation. This idea is the haptic equivalent to the simplification of graphic objects when they are away from the user; when two objects have large contact areas there is no need for testing additional details.

In [107] there are two main parameters the functional ϕ that measures the volume culled by a convex simplification and the support area D. The functional ϕ is defined as the ratio between the surface deviation of the simplified piece s_a and its resolution r_a^2:

$$\phi_a = \frac{s_a}{r_a^2} \tag{3.4}$$

The support area D is obtained for each vertex as the area of the triangles adjacent to the vertex projected on the vertex tangent plane and excluding the triangles whose normal is outside the normal cone. On aggregation of two pieces the resulting support area is the minimum of the support areas while the functional ϕ is the maximum.

On collision between two elements a and b the sensation preserving algorithm computes a weighted surface deviation s^*_{ab} using the above information from the two objects. If this value is above a treshold s_0 the collision needs to be refined:

$$s^*_{ab} = \frac{max(\phi_a, \phi_b)}{max(D_a, D_b)} \tag{3.5}$$

In our voxel model we define the functional ϕ as a measures of the volume lost by considering the volume of the lower resolution node instead of the real volume:

$$\phi_a = 1 - \frac{count_{v \in a}}{2^{3L}} \tag{3.6}$$

The support area for the volume model is computed by evaluating the surface around a Surface Voxel and by taking the minimum while aggregating the voxels into the octrees. First we compute the normal of the Surface Voxel and then we project every Surface voxel that is in the neighborhood of the current voxel. Figure 3.4 shows some cases of support area computation.

When the volume model is obtained from a triangular mesh it could be possible to compute the support area of the voxel using the same algorithm of [107].

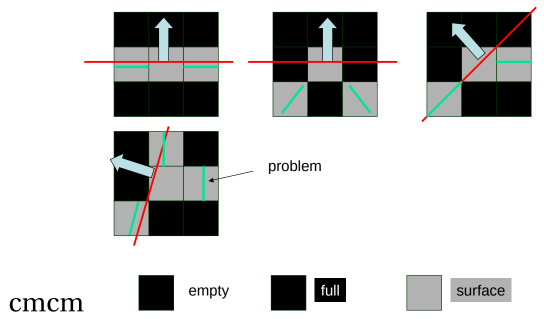


Figure 3.4: Various cases for the computation of the voxel support area

3.3 Haptic Collision Response

This section describes the collision response algorithm and the overall structure of the 6DOF haptic algorithm. First we breifly review the reference algorithm of [144], then follows a description of the collision resolution algorithm adopted.

3.3.1 Review of McNeely Approach

The Voxel Point Shell library introduced in [144] computes the Collision Detection by testing every surface point of the probing object against the world voxel model. Each resulting contact point is described by the world voxel center $\mathbf{q}$, the point over the shell $\mathbf{p}$ and its normal $\mathbf{n}$. The depth information of the world voxel is not available. From this information the force contribution of the contact point is computed using a Tangent Plane Force Model. This force model has a direction as the normal $\mathbf{n}$ and a modulus proportional to the distance d from $\mathbf{p}$ to the plane passing by $\mathbf{q}$ with normal $\mathbf{n}$. If d is positive the contact is discarded.

This algorithm has the limitation of not using the real depth of the world voxel in the force computation. Indeed the modulus of the force contribution of each contact has a maximum of $s\sqrt{3}/2$.

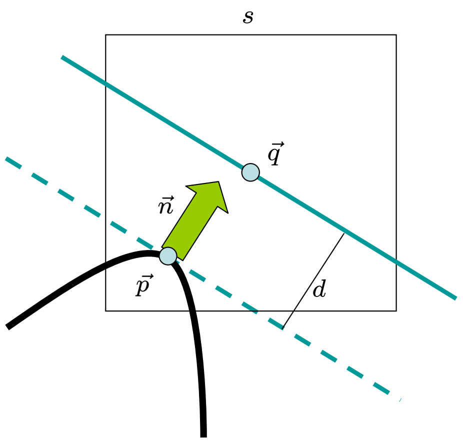


Figure 3.5: Tagent Plane Force Model

The resulting force is applied to a Dynamic Object and to the device through Virtual Coupling. The Virtual Coupling is described as a damped string that connects the Haptic Tool to the Dynamic Object. The contact forces are applied to the Dynamic Object and the force and torque are sent back to the user through the spring. For simplicity the Dynamic Object has the Inertia of a sphere.

To prevent the penetration of the world voxel model and the Dynamic Object the algorithm poses a limit to the spring displacement that is equivalent to posing a limit in the maximum force applied by the spring. The displacement is limited to $s/2$, a condition that in the case of a single voxel in contact prevents the penetration. The algorithm addresses also the problem of instabilities caused by the effective stiffness perceived by the net contributions of the contact points. For this reason the maximum contact force is clamped at a virtual stiffness of N (e.g. 10). An additional feature of this algorithm is the pre-contact breaking force that is a force that slows down the movement of the Dynamic Object when it is near the surface. In particular it is applied when the point shell collide with the Proximity voxels.

3.3.2 Overview

In this work we have decided to use a Virtual Coupling approach with Simultaneous collision resolution and impulsive collision response. The design choices can be understood inside the overall 6DOF options in Figure 3.1. We have decided to use the Virtual Coupling against the Direct Rendering because it provides a smoother behavior and also it allows to provide a spring-like force feedback when the Collision Detection and Dynamic Simulation get slower.

3.3.3 Collision Response

The collision detection algorithm described above is extremely responsive and it can be applied at almost haptic rates. The collisions are managed using simultaneous handling because the chronological approach would require a rewinding of the simulation and subsequent collision detection tests. The collision response receives a set of contact points and computes a set of impulses to the Dynamic Object to prevent the penetration. The collision with the higher depth of penetration is selected for the response and the impulse is computed by posing a condition to the velocities of the two contact point after the collision response. Although this approach is based on an impulsive resolution [98], the way contact points are resolved takes its ispiration from [56], allowing a fast resolution of the contacts.

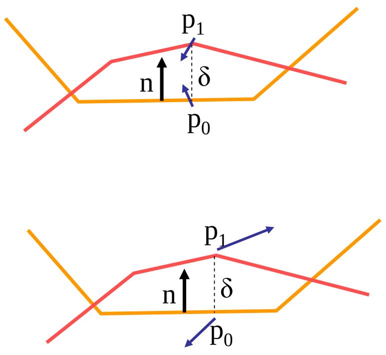


Figure 3.6: Separating and Incoming Contact Pairs

The contact response receives a list of contact pairs each described by the two points, the normal, relative velocities and the penetration depth, that is negative if the two elements penetrate. The algorithm selects the contact pair that has the biggest depth and it is not a separating pair. When two contact points has separating relative velocities they can be discarded because in the next integration step their collision would be probably resolved. When a pair is selected the algorithm resolves the conflict using an impulse that imposes a separating velocity condition in the next integration step [8] (CHECK). Such condition can be computed without friction or by taking into account the friction of the two surfaces depending on the chosen model. The impulse applied is the following:

$$j_n = \frac{-(1+\epsilon)u_n}{1/m_a + 1/m_a + 1/I_a||\mathbf{r_a}||^2 sen^2\theta_a + 1/I_b||\mathbf{r_b}||^2 sen^2\theta_b} \tag{3.7}$$

The impulse computed above will be applied to the object in the integration step, but more contact pairs could be still conflicting. Instead of applying immediately the impulse to the body and computing again the collision we continue to use the set of previous collision pairs and we apply the computed impulse to update the velocity of the body. This update allows to discard most of the contact pairs around the one computed in the previous step because of the separating velocity, and we can now start resolving the next most penetrating contact pair. This operation is repeated until there are not separating number of pairs or a maximum number of iterations is reached. The result of the collision response is the cumulative impulse that is then applied to the original body state. The Figure 3.7 shows the four steps in the case of a collision resolved by two steps of the presented algorithm, while Figure 3.8 is a snapshot from the simulation with the visualization of the collision points and the two impulses used for the collision resolution.

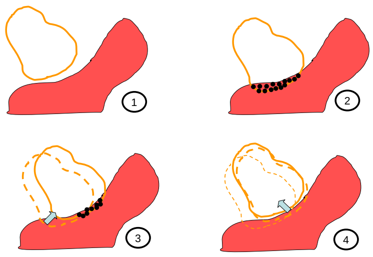
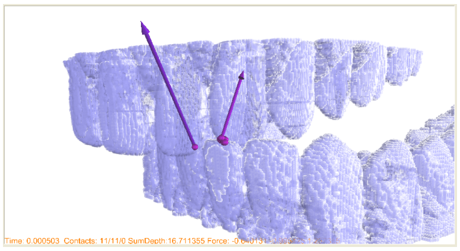


Figure 3.7: Example of collision response performed by two steps of the algorithm. The black dots are the collision pairs obtained from the voxel collision detection. The arrows represent the impulses computed at each step.



```
Time: 0.000503  Contacts: 11/11/0 SumDepth:16.711355 Force: -0.640131 -0.996 [illegible]
```

Figure 3.8: A snapshot of the collision detection and response of the algorithm with two impulses generated for the resoltution

3.4 Discussion

The 6-DOF Haptic Rendering algorithm presented above improves over the reference Voxel Volume algorithm by removing the requirements on the sampling of surface points and also by avoiding any mean based operation aimed at the simplification of the collision response problem.

The main open issue with this algorithm is the handling of deep penetrations that can happen in certain circumstances and also the complete handling of a sensation preserved optimization that could improve the performance of the algorithm itself.

Chapter 4

Benchmarking Framework for 3DOF Haptic Rendering

The benchmarking of Haptic Rendering algorithm has been usually performed using ad-hoc evaluation and user based assessment. A possible solution for the standardization of benchmarking for Haptic Rendering algiorithms is to use a set of interaction trajectories performed over a known geometric model and compare them with the exploration of real objects

4.1 Benchmarking

Haptic rendering systems are increasingly oriented toward representing realistic interactions with the physical world. Particularly for simulation and training applications, intended to develop mechanical skills that will ultimately be applied in the real world, fidelity and realism are crucial.

A parallel trend in haptics is the increasing availability of general-purpose haptic rendering libraries [36, 3, 128], providing core rendering algorithms that can be re-used for numerous applications. Given these two trends, developers and users would benefit significantly from standard verification and validation of haptic rendering algorithms.

In other fields, published results often Şspeak for themselvesŤ Ű the correctness of mathematical systems or the realism of images can be validated by reviewers and peers. Haptics presents a unique challenge in that the vast majority of results are fundamentally interactive, preventing consistent repeatability of results. Furthermore, it is difficult at present to distribute haptic systems with publications, although several projects have attempted to provide deployable haptic presentation systems [36, 55].

Despite the need for algorithm validation and the lack of available approaches to validation, little work has been done in providing a general-purpose system for validating the physical fidelity of haptic rendering systems. Kirkpatrick and Douglas [80] present a taxonomy of haptic interactions and propose the evaluation of complete haptic systems based on these interaction modes, and Guerraz et al [57] propose the use of physical data collected from a haptic device to evaluate a userŠs behavior and the suitability of a device for a particular task. Neither of these projects addresses realism or algorithm validation. Raymaekers et al [114] describe an objective system for comparing haptic algorithms, but do not correlate their results to real-world data and thus do not address realism. Hayward and Astley [62] present standard metrics for evaluating and comparing haptic devices, but address only the physical devices and do not discuss the software components of haptic rendering systems. Similarly, Colgate and Brown [19] present an impedance-based metric for evaluating haptic devices. Numerous projects (e.g. [44, 137]) have evaluated the efficacy of specific haptic systems for particular motor training tasks, but do not provide general-purpose metrics and do not address realism of specific algorithms. Along the same lines, Lawrence et al [83] present a perception-based metric for evaluating the maximum stiffness that can be rendered by a haptic system.

This paper addresses the need for objective, deterministic haptic algorithm verification and comparison by presenting a publicly available data set that provides forces collected from physical scans of real objects, along with polygonal models of those objects. We also perform several quantitative analyses on a variety of haptic rendering algorithms that do not depend on real-world data, assessing intrinsic geometric error and relative performance.

We present several applications of this data set and several standardized techniques and metrics for evaluating haptic algorithms:

- *Evaluation of realism*: comparing the forces generated from a physical data set with the forces generated by a haptic rendering algorithm allows an evaluation of the algorithmŠs fidelity.
- *Debugging of haptic algorithms*: identifying specific geometric cases in which a haptic rendering technique diverges from the correct results allows the isolation of implementation bugs or scenarios not handled by a particular approach, independent of overall accuracy.
- *Performance evaluation*: Comparing the computation time required for a standard set of inputs allows objective comparison of the performance of rendering algorithms.
- *Comparison of haptic algorithms*: Running identical inputs through multiple rendering algorithms allows identification of the numeric strengths and weaknesses of each.

4.2 Benchmarking Framework for 3DOF

4.2.1 Data acquisition

Haptic rendering algorithms typically have two sources of input: a geometric model of an object of interest, and real-time positional data collected from a haptic interface. The output of this class of algorithms is typically a stream of forces that is supplied to a haptic interface. Our goal is to compare this class of algorithms to real-world data, which thus requires: (a) collecting or creating a geometric model of a realworld object and (b) collecting a series of correlated forces and positions on the surface of that object.

We have constructed a sensor apparatus that allows the collection of this data. Our specific goal is to acquire data for haptic interaction with realistic objects using a hand-held stylus or pen-like device (henceforth called Şthe probeŤ). We use the HAVEN, an integrated multisensory measurement and display environment at Rutgers, for acquiring measurements interactively, with a human in the loop.

In previous work [108, 109], we acquired such measurements using a robotic system called ACME (the UBC Active Measurement facility). This robotic approach has many advantages, including the ability to acquire repeatable and repetitive measurements for a long period of time, and the ability to acquire measurements from remote locations on the Internet. However, our current goals are different, and a handheld probe offers a different set of advantages that are important for evaluating interaction with a haptic device.

First, it measures how a real probe behaves during natural human interaction, and therefore provides more meaningful and ecologically valid data for comparison. This is important, because the contact forces depend in part on the passive task-dependent impedance of the hand holding the probe, which is difficult to measure or to emulate with a robot arm. Second, the dexterity of robot manipulators available today is very poor in comparison with the human hand. Furthermore, acquiring measurements in concave regions or near obstacles using a robot is very difficult, but is easy for a human.

We acquired three types of measurements for each object in our data repository:

1. The object's 3D shape
2. Motion of the probe tip relative to the object
3. The force on the probe tip during contact

We describe these measurements in the remainder of this section, in reverse order. Force data are acquired using a custom-designed hand-held probe built around a Nano17 6-axis force/-torque sensor Figure 4.1 (ATI Industrial Automation, Apex, NC, USA). The reported spatial resolution of the force sensor is as follows (the z-axis is aligned with the axis of the probe): Fx,Fy 1/320 N; Fz 1/640 N; Tx,Ty 1/128 Nmm; Tz 1/128 Nmm.

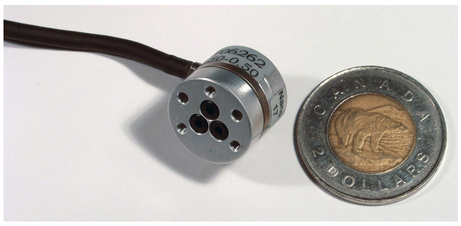


Figure 4.1: The sensor used to acquire force and torque information, alongside a coin to indicate scale

A replaceable sphere-tipped Coordinate Measuring Machine (CMM) stylus is attached to the front face of the force sensor, and a handle to the rear, allowing a user to drag the probe tip over the surface being measured. The interchangability of the probe tip is important, since the curvature of the contact area kinematically filters the probe motion and thus impacts the acquired data.

As the surface is being probed, the force/torque measurements from the Nano17 are sampled at 5 kHz using a 16-bit A/D converter (National Instruments,Austin, Texas, USA). The static gravitational load due to the probe tip is compensated for based on the measured orientation of the probe. The force and torque measured at the force sensor is transformed to the center of the probe tip to compute the contact force on the tip.

In addition to measuring force/torque, the probe's motion is tracked to provide simultaneous position data. The probe is tracked using a six-camera motioncapture system (Vicon Peak, Lake Forest, CA, USA). Several small retroreflective optical markers are attached to the probe, allowing the camera system to record and reconstruct the probe's position and orientation at 60 Hz. The position resolution of reconstruction is less than 0.5mm.

The object being measured is also augmented with optical tracking markers, so the configuration of the probe with respect to the object is known even when the user moves the object to access different locations on the surface. The object is scanned with a Polhemus FastScan laser scanner (Polhemus, Colchester, VT, USA) to generate a mesh representation of the object's surface. The manufacturer reports an accuracy of 1mm for the surface. A water-tight triangular mesh is extracted from the scans using a fast RBF method. The location of the optical tracking markers are included in the scan to allow registration of the surface geometry with the motion capture data acquired during contact measurement. Figure 4.2 shows an example data series acquired with our setup. The full data set is available in the public repository.

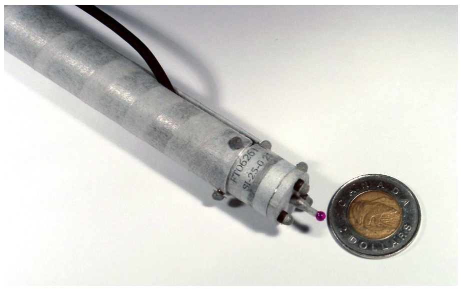


Figure 4.2: Our data acquisition system couples a custom handle and a small scanning probe with a force/torque sensor

Our initial scanning effort has focused on rigid objects with minimal friction, to simplify the analysis to the static case and to focus on normal forces.

4.2.2 Algorithm Evaluation

This section will describe the evaluations we perform on each haptic rendering system, using the data described in the previous section. The process can be summarized in three stages:

1. Post-processing of the physical data to allow direct comparison to haptic trajectories.
2. Processing of an acquired trajectory by the haptic rendering algorithm that is being evaluated.
3. Computation of performance metrics from the output of the haptic rendering system.

Figure 4.3 summarizes this process.

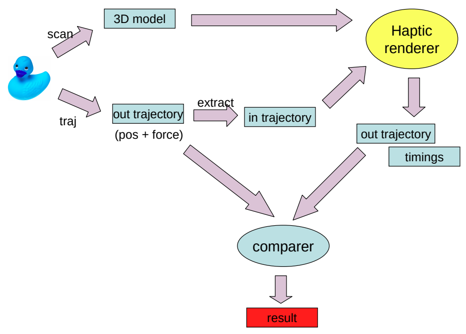


Figure 4.3: An overview of our data processing and algorithm evaluation pipeline. An object is scanned, producing a 3D geometric model and an out-trajectory. An in-trajectory is synthesized from this out-trajectory and is fed as input to a haptic rendering system, which produces force information and (for most algorithms) a new out-trajectory, which can be compared to the physical scanning data.

Data Post Processing

The haptic rendering algorithms on which we have performed initial analyses are penalty-based: the virtual haptic probe is allowed to penetrate the surface of a simulated object, and a force is applied to expel the haptic probe from the object. A physical (real-world) probe scanning the surface of a physical object never penetrates the surface of the object. Therefore a virtual scanning trajectory is not expected to be identical to a physical trajectory, even if the intended probe motions are identical. We therefore perform a post-processing step that - given a physical scanning trajectory - generates a sub-surface trajectory that would correspond to a correctly-behaving haptic simulation. This allows a direct comparison of a trajectory collected from a haptic simulation and the ideal behavior that should be expected from that simulation.

We refer to an ideal trajectory (one in which the probe never penetrates the surface of the object) as an "out-trajectory", and a trajectory that allows the probe to travel inside the object as an "in-trajectory". Figure 4.4 demonstrates this distinction.

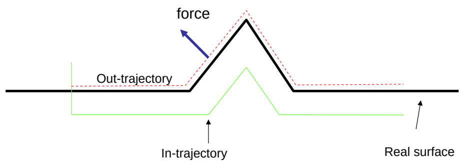


Figure 4.4: An "out-trajectory" represents the path taken by a physical probe over the surface of an object; a haptic rendering algorithm typically approximates this trajectory with an "in-trajectory" that allows the probe to enter the virtual object.

The penetration depth (the distance between the in- and out-trajectories) of a virtual haptic probe into a surface is generally dependent on an adjustable spring constant, which is an input to the algorithm and is reported along with our results. We choose a spring constant empirically, to provide the maximum stable stiffness for haptic rendering. Typically, penetration depth and the resulting penalty force are related to this spring constant according to Hooke's Law:

$$\mathbf{f_p} = -k\mathbf{x} \tag{4.1}$$

Here $\mathbf{f_p}$ is the penalty force vector, k is the scalar stiffness constant, and $\mathbf{x}$ is the penetration vector (the vector between the haptic probe position and a surface contact point computed by the haptic rendering algorithm). We use this relationship to compute a corresponding in-trajectory for a physically-scanned out-trajectory.

Each point in the sampled out-trajectory is converted to a corresponding point in the in-trajectory by projecting the surface point into the object along the surface normal, by a distance inversely proportional to the chosen stiffness (for a given normal force, higher stiffnesses should result in lower penetration depths):

$$\mathbf{p}_{in} = \mathbf{p}_{out} - |\mathbf{n}|k \tag{4.2}$$

Here $\mathbf{p}_{in}$ and $\mathbf{p}_{out}$ are corresponding in- and out-trajectory points, $|\mathbf{n}|$ is the surface normal, and k is the selected stiffness constant. This relationship is illustrated in Figure 4.5. Each in-trajectory point is assigned a timestamp that is equal to the corresponding out-trajectory point's timestamp.

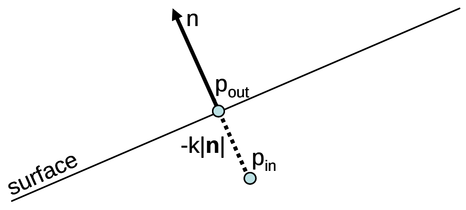


Figure 4.5: Computation of an in-trajectory point from a sampled out-trajectory point.

Following this computation, the in-trajectory corresponding to a physical out-trajectory is the path that a haptic probe would need to take in a virtual environment so that the surface contact point corresponding to that haptic probe path follows precisely the sampled out-trajectory.

Trajectory processing

The input to a haptic rendering algorithm is typically a geometric model of an object of interest and a series of positions obtained from a haptic interface. For the present analysis, we obtain a geometric model from the laser-scanning system described above, and we present a stream of positions - collected from our position-tracking system - through a "virtual haptic interface".

Given an in-trajectory computed from a physical out-trajectory, we can thus simulate a virtual haptic interaction with an object, which will produce a stream of forces and - in the case of many haptic rendering algorithms - a new out-trajectory, representing the path that a virtual contact point traveled on the surface of the virtual object.

The computational complexity of this simulation is identical to the case in which a haptic interface is used interactively, allowing assessment of computational performance in addition to algorithm output.

Metric extraction

Each time an in-trajectory is fed through a haptic rendering algorithm, producing a stream of forces and surface contact point locations, we collect the following evaluation metrics:

- *Output force error*: the difference between the forces produced by the haptic rendering algorithm and the forces collected by the force sensor. This is summarized as a mean-squared-Euclidean-distance, where N is the number of samples, Fp is the physically-scanned normal force at each point and Fr is the rendered normal force at each point:
- *Output position error*: the difference between the surface contact point position produced by the haptic rendering algorithm and the physically sampled out-trajectory. This can also be summarized as a mean-squared-Euclidean distance, although we have found that it is more valuable to collect the cases that exceed a threshold instantaneous error, representing "problematic" geometric cases.
- *Computation time*: the mean, median, and maximum CPU times required to a compute a surface contact point and/or penalty force.

4.2.3 Results

We performed the analyses discussed above on virtual representations of objects that were scanned as discussed above, using two haptic rendering algorithms: a public-domain implementation [36] of the Haptic Proxy algorithm [147], a brute-force nearesttriangle algorithm, and a rendering scheme based on voxel sampling [144]. We have released our implementations of the latter approaches along with the data discussed in this paper.

This section describes the results obtained from these experiments for three geometries:

- A simple plane, using real scanning data. This is the most straightforward case for illustrating our analysis techniques.
- A series of synthetic Şperfect spheresŤ Ű triangulated at different resolutions Ű for which we generated synthetic out-trajectories. This allows us to assess the impact of mesh triangulation on haptic rendering.
- A synthetic geometry designed to illustrate and quantify a geometric anomaly that is problematic for several common rendering schemes.

These results are presented as examples of the types of possible analyses; more complex models and the corresponding analysis results will be available on a public data repository publicly available at: `http://jks-folks.stanford.edu/haptic_data/`

Table 4.1 presents results obtained from synthetic spheres of radius 0.001 meters, for which we generated a synthetic out-trajectory that precisely follows the surface of the sphere. Friction was neglected for this analysis. These results demonstrate the powerful impact of mesh refinement on haptic rendering accuracy, independent of computational performance (which varies only slightly among the spheres).

Triangles	MSE (N)	Time (s)
48	0.5018370	0.000017320
224	0.1140975	0.000021231
960	0.0577005	0.000020952
3968	0.0297466	0.000021790
16128	0.0165954	0.000022349

Table 4.1: Results obtained from an analysis of haptic rendering using a Proxy algorithm on a series of progressively more refined synthetic spheres

Triangles	MSE (N)	Time (s)
2	2.885208	0.209460
32	2.885208	0.201283
200	2.885208	0.198253

Table 4.2: Results obtained from an analysis of haptic rendering using a Proxy algorithm on a series of progressively more refined planar meshes

Table 4.2 presents a comparison of a haptic interaction with a virtual plane and a physical interaction with a real planar object We also vary the number of triangles used to represent the plane, and the presented results confirm that the rendering accuracy is independent of the planeŠs triangulation, as one would expect for a planar surface.

Figure 4.6 illustrates a problematic geometry that can be captured by our analysis approach. In this case, for certain stiffness values and angles of extrusion (i.e. Şbump sharpnessŤ), the surface contact point produced by the Proxy algorithm becomes ŞstuckŤ on the bump, producing an incorrect trajectory that misrepresents object geometry. Our approach allows a rapid evaluation of this geometry using a variety of synthetic models and a variety of algorithmic parameters (friction values, stiffnesses), allowing quantification of such problematic cases for particular renderer implementations.

Table 4.3 uses this geometry to compare three rendering schemes in terms of force error and computation time. We see that the voxel scheme incurs a high overhead in initial computation (for volume discretization), and that the Proxy algorithm produces high mean errors due to the geometric anomaly illustrated in Figure 4.6.

Algorithm	Mean Time (s)	Init Time (s)	MSE (N)
Proxy	8.1e-6	0	0.298
Potential	12.3e-6	0	0.0417
Voxel	3.6e-6	0.270	0.0949

Table 4.3: Comparison of several algorithms processing the geometry illustrated in Figure 4.6
.

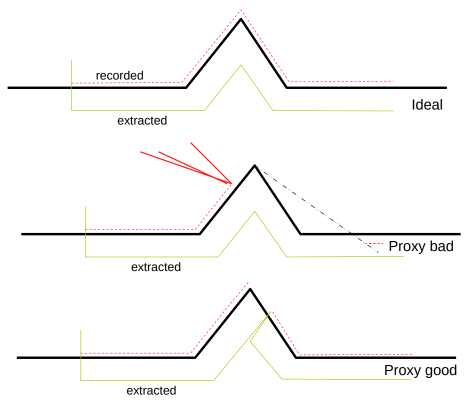


Figure 4.6: Our evaluation approach is able to identify and quantify failure cases for the Proxy algorithm

4.2.4 Discussion of the results

We have provided several analyses that are independent of the specific data sets used and the specific haptic rendering algorithms that were evaluated. Similar analyses could be applied to a wide variety of data sources and rendering systems.

An obvious application of this analysis is to assess the realism of a particular haptic rendering system and to approximately bound the difference between the forces experienced by a user through a haptic interface and the forces the user would experience performing the same interactions with a real object. This analysis can also be used to compare haptic rendering algorithms more objectively: if one algorithm consistently produces a lower force error relative to a real data set than another algorithm, it is objectively Şmore realisticŤ by our metrics.

This approach has an application not only in evaluating published rendering systems, but also in debugging individual implementations. Debugging haptic rendering systems is notoriously difficult relative to debugging other computer systems, due to the hard-real-time constraints, the nondeterminism introduced by physical devices, and the difficulty of reliably replicating manual input. Our approaches and our data sets allow a developer to periodically test a haptic rendering system via a series of objective evaluations, and thus rapidly identify problems and isolate the changes that caused them.

We have also provided an objective series of input data that can be used to evaluate the computational performance of an algorithm. In this context, our data sets and analyses provide a Şhaptic benchmarkŤ, analogous to the rendering benchmarks available to the graphics community, e.g. 3DMark. Computational performance of a haptic rendering system can vary significantly with input, but it is difficult to describe and distribute the input stream used to generate a performance analysis result. By providing a standard data series and a set of reference results, we present a performance benchmark that authors can use to describe algorithmic performance. This is particularly relevant for objectively presenting the value of optimization strategies for rendering and collision detection whose primary value may lie in performance improvements. Performance results are still dependent on the platform used to generate the results, but this information can be reported concisely along with results.

This approach is not necessarily a complete description of a haptic rendering algorithmŠs quality or performance. Algorithmic performance and even results are expected to vary somewhat when collected with a user and a physical device in the loop, and no set of reference data can completely capture all possible cases that may have particular impacts on various rendering algorithms. But we propose that a standard approach to haptic rendering analysis and standard data series will significantly enhance the quality and objectivity of haptic rendering system evaluation.

4.3 Discussion

Acknowledgments

I would like to acknowledge the people that have contributed to this research, in particular Dan Morris and Federico Barbagli from Stanford University, Timothy Edmunds and Dinesh K.Pai from Rutgers University. Also support for this work was provided by NIH grant LM07295, the AO Foundation, and NSF grants IIS-0308157, EIA-0215887, ACI-0205671, and EIA-0321057.

Chapter 5

Integrating Haptic interaction on the Web

This chapter introduces a software framework for the development of Haptic application on the Web called HapticWeb. The framework provides all the elements for the fast prototyping of applications hiding many of the programming complexities and allowing the developer to focus perceptual or user interface aspects. HapticWeb is addressed to both perceptual scientists that want to create perceptual experiments, to students for the creation of haptic games and for everyone who wants to create haptic applications.

5.1 Rationale

The recent developments of haptic interfaces and haptic software, both in terms of performance and cost, make more pressing the necessity of creating tools for the easy development and deployment of haptic enabled applications. The goal is to improve the teaching and experimentation of haptics among students, the rapid prototyping of applications and the construction of experiments for the validation of the perceptual aspects. Additional motiviation is provided by the number of applications in which haptics can be applied such as virtual museums, virtual prototyping, and training.

The validation of perceptual aspects relative to haptics has been usually addressed by developing ad-hoc applications based on the GHOST libraries by Sensable, or by using generalized tools for haptic experiments like Enchanter [46]. The current research on haptics has been integrated with other sensorial modalities like in the ENACTIVE Network of Excellence, with the necessity of more flexible frameworks for experimentation in which is possible to easily integrate and test haptic, vision and sound.

The idea of a haptic virtual museum has been developed by the Pureform project [88] that enables the user to haptically and visually explore virtual statues obtained from real museums. This project has been presented in real museums and in special events using a hand exoskeleton and stereographic visualization. Currently the Web version of the Pureform museum is visual and it could be enhanced with the interaction provided by desktop haptics.

Although the idea of the Web 3D has not yet flourished as a set of working standards there are some specific fields in which 3D application are spreading on the Web. Among these fields the case of Virtual Manuals is one of the most prominent. A Virtual Manual is a hypermedia in which text and 3D models are integrated for documenting an industrial entity in assembly or mainteinance. Low cost haptic interfaces could be used for improving the exploration and manipulation of such 3D models, and in this perspective haptic extension of Virtual Manuals would be effective.

5.1.1 Multimodal Systems on the Web

The experience provided by the current Web is limited to multimedia content, with the dynamic download of both the application and the multimedia resources. The general concept of a multimodal system is currently addressed by the W3C Multimodal Architecture and Interfaces [11]. The latest Draft of this document is a general description of a flexible architecture for multimodal systems in which the different components are managed by a shared runtime environment, a solution that is too general for being supported by a practical implementation.

The integration of haptics with the Web has been addressed by integrating a VRML [143] browser with haptic rendering with the addition of extension nodes [106, 99]. One of the most promising solutions based on VRML is H3D by SenseGraphics [3] that provides a X3D implementation based on Sensable's OpenHaptics toolkit [70]. The first limitation of these systems is the support of a specific family of devices and interaction modalities. The other aspect that we want to point out is the dependency on the VRML format. This is cleary a choice that favours compatibility and standardization although there are no standards for haptic nodes. The use of VRML decides the interaction mode and the logical structure of the application.

Existing haptic libraries can be organized mainly in two groups. First there are the low level libraries that are device specific and allow a direct communication with the device [38]. Then there are the high level libraries that provide haptic rendering features extended with graphical visualization [115], although some of them are device specific depending on the Sensable's OpenHaptics toolkit. One of the most promising is the CHAI3D library [36] that supports different devices, has an Open Source license and provides a haptic scenegraph for the application construction. Still the development of the application requires a fair amount of C++ knowledge. When looking at the commercial libriaries the Reachin API [127] is one of the most capable both in terms of haptic rendering, support of devices and easiness of development.

5.1.2 Multirate in Virtual Reality

An important aspect in the development of Virtual Reality application is the management of the task associated with the interaction modalities. The primary element is the visualization running at 50Hz, with one or more channel depending on the type of visualization, from two of the stereo up to six in a CAVE. Then we can add the sound channel with a sampling rate of 44kHz, and dynamic simulation based on physics running at around 200Hz. The introduction of haptics requires an additional channel running at 1kHz and finally we can have additional sources depending on the presence of position trackers and networking.

The data sources described above pose problems to the developer both in terms of multitaksing and of the multiple representations of the same object. An object in the Virtual Environment requires multiple representation with possibly duplicating geometry and status information as shown in Figure 5.1.

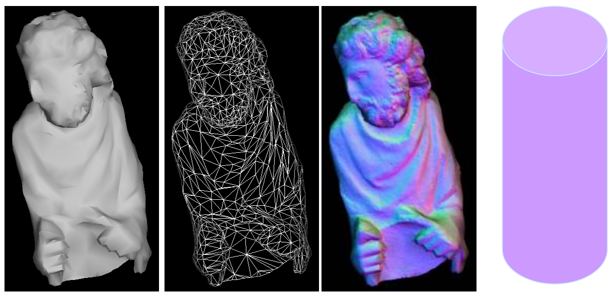


Figure 5.1: Multiple representations of the same object

A complete Virtual Reality system should be able to manage all these data sources and hide most of the detail to the developer. The complexity of management of each of these channels depends on its maturity. For example sound management is mostly hidden to the developer through spatialized 3D sound APIs like OpenAL or synthesized audio tools like PureData. When dealing with haptics, instead, the developer needs still to specify the forces to the contact point and compute them at 1kHz with the fine control over the chosen algorithm. Apart the specific behavior decided for a modality the developer should be able to view an object as a single entity with all its modalities and give the possibility to integrate such object with others distributed in a network as show in Figure 5.2, a topic discussed by the author in [118].

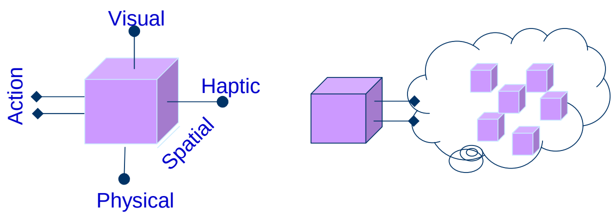


Figure 5.2: Schematic representation of a multimodal entity and its connection to other entities

The way the developer manages all the complexity of Virtual Reality application is reflected also in the structure of the application and in the programming model. VRML manages most of the complexity by defining nodes that expose events and allowing the developer to connect such events in a network. This solution is powerful but limits the possibilities of the developer by defining a single programming model.

5.2 Architecture

HapticWeb is a script based framework for the development of haptic enabled application that are almost independent from the specific haptic interface [1] and it provides enough flexibility for its extension with additional modules, a feature that is fundamental for the creation of complete multimodal applicationss. This framework completes the initial work done by the Author in [116], where the core features and the device independence where already present and the main differences were the use of the Lua scripting language and the lower quality both on the graphics and the haptic rendering.

The current HapticWeb system is based on the Virtual Reality engine eXtreme Virtual Reality (XVR) [23, 22] constitued by a fast and simple bytecode virtual machine, a set of core 3D graphics features and a deployment mechanism that allows an easy distribution of the application on the Web. HapticWeb provides new modules for haptic interfaces, haptic rendering and dynamic simulation not available natively in XVR, and also a set of higher level classes that simplify the development of multimodal applications. Figure 5.3 shows the HapticWeb architecture with the core XVR modules and the modules provided by HapticWeb [14].

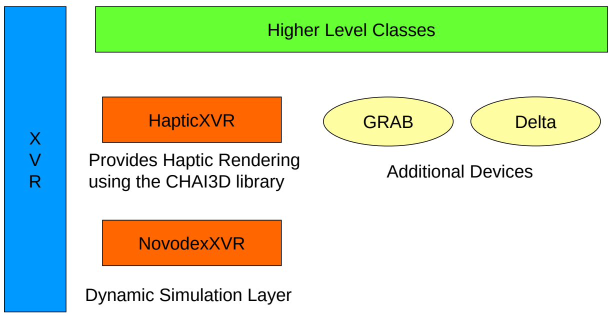


Figure 5.3: The architectural view of HapticWeb

The HapticWeb application is described by a scripting language specialized for 3D graphics that is able to control the different modalities through an extensive object model, and at the same time it allows the low level control of the graphics through OpenGL commands. After the download and the initialization phase the developer has complete control over the graphics and haptic rendering, and he is required to define the application logic and the interaction between the different modalities. When the developer has prepared the script program and the associated multimedia resources, he publishes it on the Web inside the Web page, along with archives containing the resources. The application is executed inside the Web Browser using a plugin for 3D graphics that loads the program, the associated 3D models, and executes the application. The HapticWeb program is associated with a specific version of the runtime engine and the plugin automatically downloads the requested version. With this approach we separate the update of the application, done by downloading the code from the Web at each execution, and the update of the HapticWeb runtime automatically performed by the plugin.

The Figure 5.4 shows an interactive session with HapticWeb inside the Web Browser. In this case a 3D model obtained from a laser scan was explored with a PHANTOM Omni [92] and some

[1] Being based on CHAI3D, HapticWeb supports most commercial kinesthetic devices based on impedance control

haptic parameters were accessible from the Web page.

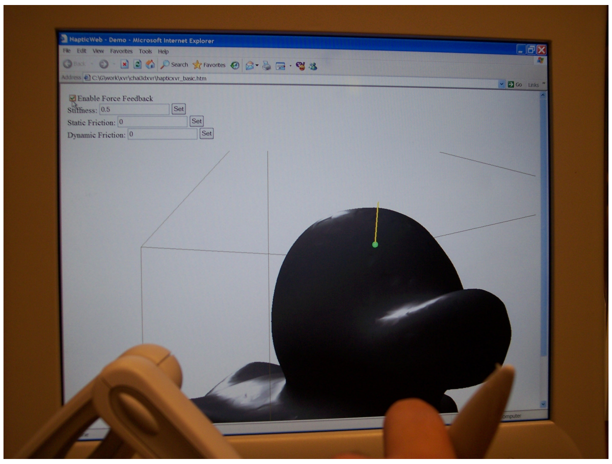


Figure 5.4: A picture that shows an interactive session with HapticWeb

5.2.1 XVR

The HapticWeb application is deployed on a Web site as a compiled program and a set of multimodal resources that can be downloaded from the network and made accessible from a Web page. The execution of the program takes place inside a Web Browser's plugin that provides the integration with the Web page and the network. A virtual machine evaluates the program in the form of a bytecode representation. The bytecode representation for the distribution of Web applications has been successful in commercial systems like Java and Macromedia Flash, because of the compactness of the representation, support for multiple platforms and security. Additionally the bytecode adopted for this project has been tailored for 3D graphics applications.

The execution environment, the Web Browser's plugin, and the virtual machine for this system are provided by the XVR engine developed by PERCRO laboratory and presented in [22]. The XVR platform has been used in many Virtual Reality projects running both on the Web or inside highly immersive installations, and it is also used in a Virtual Reality course.

A HapticWeb program is written using the XVR scripting language, an object oriented scripting language specialized for 3D graphics, with a C-like syntax that resembles current shading languages. Its specialization is focused on the efficient management of vectors, such as support for the swizzle operator.

XVR applications are organized around callback functions invoked during specific events and system loops. In general it is possible to describe a multimodal application as a multirate program, with a common logic that coordinates the different modalities. The logical loops of the XVR platform are shown in Figure 5.5 but for making the structure of the application simpler only two of them are explictly associated with callbacks: the graphics loop and a generic timer loop, the first running at the display refresh rate and the other at 1KHz.

The XVR runtime engine provides support for 3D graphics and audio spatialization exposed as an object model accessible to the developer in the program. The developer has low level

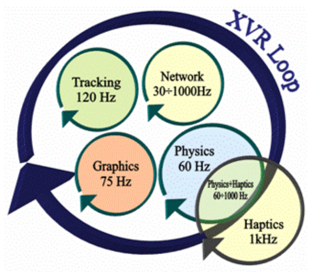


Figure 5.5: This diagram shows the loops of a typical multimodal system withing XVR

control of the graphics through OpenGL commands and at the same time the visualization of complex, multi-material and animated models generated with standard 3D modellers. The low level access provides flexibility in the generation of specific 3D elements or effects whereas the high level access provides the standard scene graph approach for the visualization of 3D graphics.

5.2.2 The CHAI Haptic library

The XVR's object model has been extended to support haptic feedback providing support for devices, haptic rendering of objects,and haptic effects, expanding in this way the range of possible applications of the XVR platform. The haptic functionalities of the HapticWeb platform are provided by the CHAI 3D haptic library [36], an Open Source effort of Stanford University for multiple haptic devices and multiple platforms. We have chosen this library because it is device independent and it provides different choices for the implementation of the haptic loop. CHAI uses a haptic scene graph for organizing objects and points of contact of the devices. In this project we have exposed most of CHAI's features to the scripting system and extended them for haptic feedback realism and expressiveness.

5.2.3 Device Access

The support of multiple haptic devices is one of the fundamental requirements for the spreading of a haptic application framework, and HapticWeb takes advantage of the variety of haptic devices supported by the CHAI3D library. The overhead for such flexibility is limited and can be measured as an additional 150K of data downloaded for supporting all of them. In a single device configuration we try to figure out which device is currently active and initialize it. Eventually, if there is no haptic device attached, a device simulated using the mouse is used. HapticWeb provides the possibility of initializing different devices by using a device URI. A device URI specifies the type of device, its identification if there are many of them and optional parameters expressed using the query part of the URI. For example a PHANTOM can be accessed using "phantom://default?mode=direct" where the mode parameter specifies use of Direct I/O if available. In the case of a remote device we can write "remote://145.22.33.44/device0". In general we suggest using the automatic device configuration because it reaches the maximum audience but in some cases a specific one could be required.

Multiple devices can be instantiated as well, and each can be associated with a tool for the force rendering of the point of contact. In the case of the GRAB device [5] with two arms, each arm is identified by a different URL and requires separate haptic rendering: “ehap://grab/left” and “ehap://grab/right”.

One of the problems in the portability of haptic applications is the difference between the devices in terms of workspace and force feedback limits. We address the problem by providing to the developer detailed information about the device, a feature that is not available in most of libraries and also we avoid any workspace scaling that affect the realism of force feedback.

5.2.4 Expressing Haptics

The haptic feedback capabilities presented so far are relative only to the standard object touching interaction that can be obtained using proxy-based algorithms, but a haptic application sometimes requires the generation of force effects for notification of events to the user or force fields for constraints and guidance. For example a needle insertion simulation could provide a force field as a haptic hint for the task. HapticWeb provides a set of force effects that can be applied as superimposed or as alternatives to the surface feedback. Each effect can be enabled explicitly or it is associated with a time duration useful for triggered effect. At the same time there is an activation volume in which the effect is active represented by a sphere. Finally the force exerted by the effects can be expressed locally or globally to the haptic point of contact.

The effects that we provide can be categorized by generic force fields and constraints (see Figure 5.6 for a schematic representation). The first group contains simple effects like spring, implicit sphere, virtual plane, friction or constant force associated with a certain bounding volume. However the constraint effects are described using a set of points, lines and triangles that specify attraction points for the haptic point of contact, with different levels of constraint strength. The resultant force from the haptic loop is obtained by exclusively choosing between the nearest active constraint and the surface feedback, and superimposing to that the other forces caused by the other active force field effects.

The use of constraints, in particular of spline based curves, can be used also for volume visualization applications as in [87], that discuss the integration of constraints with proxy algorithms.

The geometry used for the haptic feedback is typically obtained from one precomputed model, but there are cases in which we need to construct it dynamically or we have existing code that visualizes some geometry using OpenGL commands. For the above reasons HapticWeb provides a geometry capture feature that allows construction of the geometry used for the haptic rendering from the OpenGL rendering. The geometry inside the view frustum that is sent to OpenGL inside the capture region is used for updating the haptic mesh. The other application of this technique is for the specification of the haptic constraints discussed above. Every point or line sent to OpenGL inside the effect capture region is transformed into a constraint primitive[2]

[2]The geometry capture for a haptic mesh could be performed inside the haptic world semaphore, but this could completly stall the haptic rendering, for this reason is better first to disable the object from the haptic world and then perform the geometry capture operation.

The following is an example of the code necessary to render an implicit sphere by using the script code directly:

```
function OnTimer()
{
        var tool = h_tool.tool;
        tool.updatePose();

        var delta = tool.devicePosition - sphereCenter;
        var d = modulus(delta);
        if(d > sphereRadius || d == 0)
                tool.force = [0,0,0];
        else
                tool.force = sphereStiffness*(sphereRadius-d)*(delta/d);
        tool.applyForces();
}
```

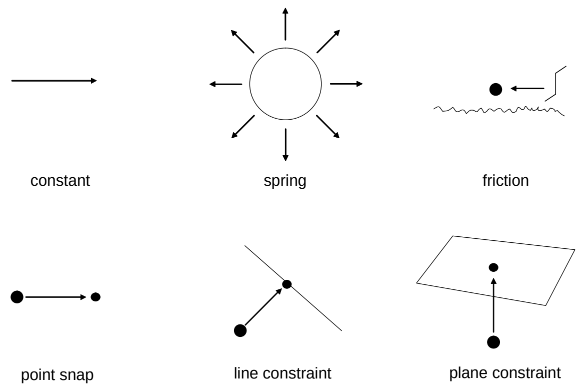


Figure 5.6: A schematic representation of haptic effects provided by HapticWeb, in the top row the field effects and in the bottom row constraint effects

5.2.5 Web Integration

The integration with the Web is not limited to the on-demand deployment model or access to networked resources. First we are going to discuss the URI based object namespace inside the haptic scene, then the interaction between the HapticWeb application and the container Web page.

Resource Namespace The integration of HapticWeb with the Web architecture is not limited to the elements discussed so far, but it has been adopted internally for the identification of resources. The multimodal entities that describe the virtual world of the HapticWeb scene can be accessed both through the scripting language through variables that refer to the objects and through Uniform Resource Identifiers (URI). The runtime framework of HapticWeb provides a namespace system for accessing entities for their identification and manipulation, the root of this namespace corresponds to the instance of the virtual environment running on the machine, and the children entities provide ways for accessing the objects contained in the VE. This resource-based approach makes it easier for both the internal script and the external Web page to identify and manipulate the entities: moreover it provides the possibiltiy to identify the resources in a remote HapticWeb environment. First we will discuss the part of the URI that identifies the current scene and then how it is possible to access the objects and their properties.

An object inside the scene can be identified using a complete URI that can be used by other HapticWeb scenes running on remote machines. This URI is made of a part that identifies the machine and the specific scene in which it is running, and then a part that identifies the object and its properties. A specific HapticWeb scene running on a machine can be identified in different ways. A first possibility is to use the Web Server as a proxy, and to encode a scene identifier in the URI, but this solution limits the direct comunication between the scenes in different machines. A simple, but effective solution is to use the host address and a port number with the pseudo-scheme "tcp" or "udp", for example: "tcp://145.3.4.5:9100/objects/A/B". This solution is limited as well in the case of firewalled hosts. A solution that overcomes the problem is to use a Peer-To-Peer layer that is capable of providing connectivity between most of the hosts. A promising solution in this direction is the JXTA protocol. The current implementation of HapticWeb provides only the solution using the pseudo-scheme "tcp" and "udp", with the possibility of intregrating the P2P approach using the C-JXTA library [139].

The objects of the scene can be organized in a hierarchy and each object associated to a name. This naming scheme constitues the representation hierarchy accessible through the "/_objects/" path. For example if an object C is child of B and B of A, then it can be accessed through "/_objects/A/B/C/". The representation hierarchy usually does not correspond to the logical hierarchy because some objects could be present only for implementation purposes. The logical hierarchy is expressed using a mechanism similar to the object identifier provided by Web pages, in which an object at any level of the representation hierarchy can be associated to a top level identifier. In the HapticWeb the logical tree is accessible using an URI starting with "/objects/".

When the object has been identified using one of the two trees, it is possible to identify one of its properties by appending the name of the property. For example, to access the position of the object C we can write "/objects/C/pos". The modification of the property value is obtained by using the query part of the URI, for example "/object/C/pos?value=1,2,3". This approach allows the user to access properties in the local scene or in a remote scene, allowing, for example, information transfer and synchronization between them. The component that manages the resource access through URI is conceptually similar to the Dynamic Property Framework described in [68].

Interaction with the Web page HapticWeb allows the creation of two types of applications: immersive or embedded, depending on the use of the visualization system. Immersive 3D environments running in full screen or with stereographic displays belong to the first type, while the latter refers to applications embedded in the Web page.

When embedded in the Web page the haptic scene interacts with the container using asynchronous events that allows exchange of data and commands with the JavaScript in the Web page. A first example of this feature is the possibility of using the Web page as a 2D user interface for the 3D world as in Figure 5.4, although this solution is only valid in contextes where immersivity in the 3D environment is not a requirement. At the same time it is possible to use the HapticWeb as a user interface for the Web page for visualizing and exploring the structure of the site or for the control of a multimedia resource as in [131].

The multimedia control is an example simple enough for understanding the possibilities of the embedding; in particular we would like to control the playback rate of a movie contained in the Web page by using the haptic device. The force feedback is generated by a spring that attracts the point of contact to the origin. The speed of the movie is controlled by the oriented distance of the point of contact to the origin, and fastest speeds correspond to higher forces. Finally the communication from the haptic scene to the movie control in the page is obtained by generating an event that is captured by the JavaScript code in the page and used for modifying the speed of the movie.

Multimodal Magazine A particular extension of the Web integration is the case of creation of hypermedia document that contain a 3D multimodal environment embedded in their page layout. Figure 5.7 shows an example of document about Pool Table's Physics that has been enhanced with the possibility of haptically experiencing the content described in the text.

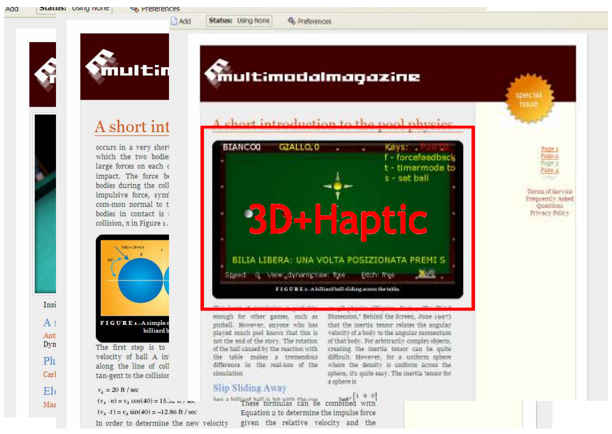


Figure 5.7: An example of integration of the HapticWeb system inside a Web page for new types of documents

5.2.6 Extensibility through Python - PYXVR

The extension of HapticWeb application with external modules requires the creation of an External DLL for the XVR engine. An alternative to this approach is PYXVR [117]. PYXVR is a special module for XVR that allows to develop XVR application, and consequently HapticWeb applications, using the Python language. There are two ways of using PYXVR, the first is to use Python for providing to XVR the access to the large number of Python modules, the second is to develop HapticWeb applications using the Python language. The integration of Python in XVR is full allowing the complete access to the XVR types and classes both internal and external, as the case of the haptic module used in HapticWeb. The architectural structure of PYXVR is shown in Figure 5.8, where is clear the dual scripting structure of PYXVR. Among the additional possibilites offered by PYXVR there are the support for multi-threading and debugging, features not yet available in XVR.

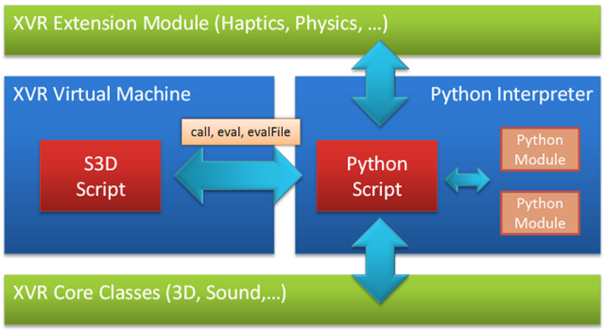


Figure 5.8: Architecture of the PYXVR system, showing the relationship between the two scripting systems and the modules

The following example is a small PYXVR application that displays a 3D grid and invokes functions from the XVR core and from the associated script. First the S3D code that invokes the Python code:

```
#include <Script3d.h>

#define ENGINE_VERSION "0141"
extern function PythonEngine;
var py;

function OnDownload(script)
{
        FileDownload("pyxvr.zip");
        FileDownload("pyxvrminimal.py");
}

function OnInit(script)
{
        LoadModule( "pyxvr_"+ENGINE_VERSION+".dll");

        py = PythonEngine();
        py.evalFile("pyxvrminimal.py");
        py.call("OnInit");
}

function DrawGrid(n)
{
        var i;
        glLineWidth(n);

        glDisable(GL_LIGHTING);
        glColor(0.5,0.5,0.5);

        glBegin(GL_LINES);
        for( i= -100; i<=100; i+=10 )
        {
                glVertex(i, 0,  100 );
                glVertex(i, 0, -100 );

                glVertex( 100, 0, i );
                glVertex(-100, 0, i );
        }
        glEnd();
}

function OnFrame()
{
        py.call("OnFrame");
}

function OnExit()
{
        OutputLN("OnExit!");
        py = Void;
}
```

Then the Python code:

```
#import rpdb2; rpdb2.start_embedded_debugger("Hello",True)
from pyxvr import *

mesh = None
pos = 0.5

def OnInit():
        global mesh
        mesh = CVmNewMesh("box.aam");
        mesh.Normalize(1)
        SetCameraPosition([0,2,-10]);
        CameraSetTarget(0,0,0);

def OnFrame():
        global mesh
        global pos
        SceneBegin();
        mesh.Draw()
        glTranslate(0,pos,0);
        XVR.DrawGrid(3);
        SceneEnd();
```

5.3 Evaluation

In this section we present some analysis of the perfomance of HapticWeb and its easiness of development.

5.3.1 Haptic Loop

The haptic loop is one of the key aspects of any haptic application because its efficient implementation provides a stable haptic interaction, usually obtained by maintaining a rate around 1kHz. The HapticWeb platform allows the developer to choose between two different approaches for the definition of the haptic loop. In the first approach we describe the haptic scene using a scene graph made of 3D models and haptic effects, and the engine manages the haptic loop in a way transparent to the program. The developer can access and manipulate the state of the objects only after obtaining a lock on the haptic scene. With this approach the resulting performance of the haptic loop is the same as of a native application, that is only limited by the precision of the timer's scheduling of the operating system.

The second approach is more flexible because it allows the developer to use both the haptic scene graph and scripting to compute the force feedback. The purpose of this feature is to allow experimentation with new haptic effects or rendering techniques not provided by HapticWeb. In general the performance difference between the scripting approach and the native haptic loop depends on the multithreading support of the scripting system. The current approach of HapticWeb is a multithreaded execution environment in which the virtual machine running the script is a shared resource associated only to one thread at a time. At each iteration on the two main loops, one for the graphics running at 75Hz and the other a generic timer running up to 1KHz, it waits for the virtual machine semaphore, allowing the execution of the corresponding callback function. In this way the scripting evaluation is equivalent to a single threaded approach that simplifies the synchronization aspect for the user, but can introduce performance problems in the haptic loop. The force computed by the scripted haptic loop is sent to a native thread that runs exactly at 1KHz and communicates it to the device's driver. In this way the delay of the timer loop is a possible source of instability although not a source of the timeout for the device.

As we are going to show later in the experiments section, the performance of a script-based haptic feedback in HapticWeb is well suited for haptic feedback only if the graphics loop is performing light computations and not blocking the haptic loop. Fortunately there is a third possibility that maintains the flexibility of the scripting and the performance of the native haptic loop. This third approach uses an *intermediate representation* or local model of the force feedback between the high level part, the script, and the low level loop, the native component. This representation has been adopted in networked haptic rendering for overcoming the problems connected to the delay (see [91]) and it can be applied to this system as well. Instead of computing and sending forces to the device from the script we compute a local surface parameterization that is used by the native loop to perform the force rendering. Two examples of intermediate representations are the *plane and probe* and the *point-to-point spring*. Figure 5.9 shows the three approaches of the haptic loop inside HapticWeb.

The script side of the haptic loop computes the collision of the probe with the surface and

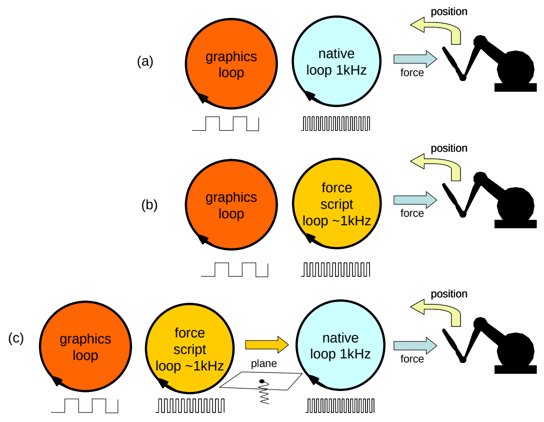


Figure 5.9: The three ways of implementing the graphic-haptic loops: (a) native only (b) script only (c) intermediate based. Each circle represents a functional loop and the oscillation gives a qualitative idea of the loop's rate

produces an oriented plane with a specified stiffness value that is sent to the native haptic loop. In this way a drop of the script loop to 500Hz does not hurt the stability of the interaction. Respect the networked case, the tight connection between the two loops allows a reduction of the artifacts associated with this intermediate representation. This approach can be obtained inside HapticWeb using two haptic scenes. The first contains all the models and a virtual point of contact, whose position is obtained from the real point of contact. The slower script-based loop computes the proxy in the standard way and it updates position and orientation of the virtual plane. The second scene, containinig the real point of contact and the plane, has a full speed haptic loop. Surface properties can be added to this local model, although other properties like textures are more diffult to add.

We have peformed some performance measurements of the haptic loop delay under various graphics loads for showing the conditions under which is possible to perform a script-only haptic loop.

5.3.2 Force Rendering Tests

The performance measurements for haptics usually requires a user assessment of the stability of the result with the consequence of introducing human related noise in the measurement. In these tests we used the benchmarking framework introduced in [119] in which the haptic feedback is computed and compared against an exploratory trajectory, over the object recorded using a force and position probe. We feed the haptic loop with positions from the recorded trajectory computing the delay in the response and the error in the force result. The measurement of the performances has been obtained using the open source Performance API (PAPI) ([86]) that provides a multiplatform access to the hardware counters of the processor for evaluating the exact timestamp and the number of floating point operations performed.

The performance test has been performed on a Pentium M 2.0GHz with an NVidia Quadro Fx Go 1400 graphics card. We took two trajectories of about 6 seconds probed over a laser scanned object with a trajectory sampling of 1kHz (see Figure 5.10). The comparison has been done between the script-based loop and the native loop under different conditions of graphics load. In Figure 5.11 we show the average haptic loop period of the script-based loop against the reference native loop with six load levels and three mesh resolutions of 3k, 64k and 137k triangles . The graphics loads level goes from a none operation for the graphics to a multipass rendering with a read pixel operation.

What happens is something that could be expected from the current solution provided by the script. While performing haptic rendering using the script, it is fundamental to keep the graphics callback loads low, which corresponds to sending a small number of OpenGL commands to the driver and to not performing operations that stall the OpenGL system. In the case of pixel reading the OpenGL driver needs to terminate the rendering while the script execution is inside the graphics loop, with the result of deeply impacting the haptic loop's performance as shown by the case “ORC”. In general the graphics loop load depends more on the way the graphics commands are sent to the OpenGL system than the absolute complexity of the scene. A graphics loop that does not create dependencies to the OpenGL performs the rendering phase outside the script callback allowing a peformant execution of the scripted haptic loop.

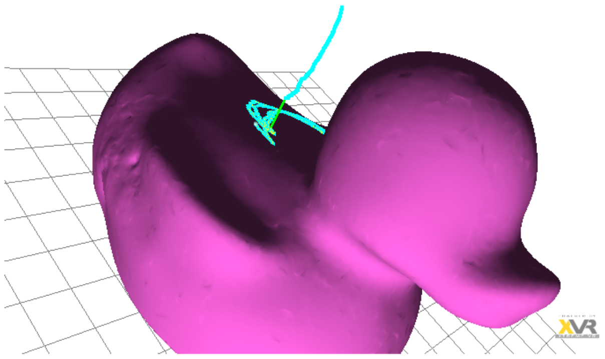


Figure 5.10: This is an example of the probed trajectory used for benchmarking the haptic interaction with the model using the various haptic loop approaches

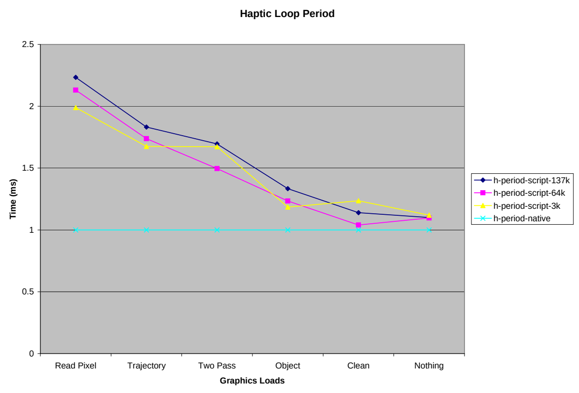


Figure 5.11: The graph shows the relation between the haptic loop period and the graphic load, comparing the native case (1ms) against the script-based loops with different resolutions of the model

We have shown that under some conditions the script only haptic loop provides stable results, and when needed it is possible to use the intermediate representation that provides a tradeoff between stability and flexibility. A first improvement is a more complex local model, but additional flexibility could be provided by a specialized script, a force shader, that is independent of the rest of the application script. This solution has some analogies with the trend of computer graphics from fixed function pipelines to shaders.

5.4 Applications

This section presents some of the applications developed using HapticWeb highlighting the feature of the framework that have been used.

5.4.1 Haptic Pool

A complete example of application using HapticWeb is the Haptic Pool, that allows to play Billiards using a haptic interface. This example integrates the dynamic simulation of the pool table with the haptic feedback using the HapticWeb framework described above. The haptic interface is used for impressing force and direction over the balls, and also for changing the point of view of the player, using the direct rendering of the forces. Figure 5.12 shows the application while playing with the GRAB device.

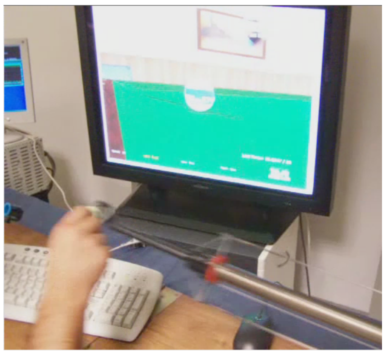


Figure 5.12: Example of the Haptic Pool application in which the GRAB device is being used

The application is enhanced with audio feedback to provide the sound of collisions between the balls with the cushions and other balls. The user decides the hit direction through the haptic interface; then by pressing a button on the device, a virtual sliding is implemented that constraints the cue to move only forward and backward along a line aligned with the hit direction and through a point $\mathbf{p}$ of the ball, that represents the hit point. Basically the force-feedback is computed as an impulse force assumed proportional to the hitting velocity, $F_{hit} = kv_{hit}$. If T is the sampling time, an impulse force $I_{hit} = F_{hit}T$, is then applied to the ball at the position $\mathbf{p}$ where the cue hits the ball, the linear and rolling initial conditions of the dynamics of the ball are given by:

$$\begin{cases} m\mathbf{v}_{in} = I_{hit} \\ I\omega_{in} = \mathbf{p} \times I_{hi} \end{cases} \tag{5.1}$$

The hit point $\mathbf{p}$ can be changed by the user through the arrow keys to implement underspinning effects, see for instance the green point in . Billiard cloth is implemented through static μ_s and dynamic friction μ_s properties, and with an additional constant force term $F_{el} = k2mg$ proportional to the ball weight, that models the dissipation of energy due to the elasticity of the cloth under the weight of the balls.

Then the free dynamics of the ball is computed to determine the evolution of position of the ball over time, until collisions either with other balls or cushions happen. In static conditions we have, indicating with R the ball radius, and by considering the moment equilibrium equation at the contact point

$$\begin{cases} v = \omega R \\ I\frac{d\omega}{dt} = -F_{el}R \end{cases} \tag{5.2}$$

while in dynamic conditions, with sliding occurring between the ball and the cloth

$$\begin{cases} m\frac{dv}{dt} = -\mu_d mg - F_{el} \\ I\frac{d\omega}{dt} = -F_{el}R \end{cases} \tag{5.3}$$

The collisions are modeled with simple geometry reflection rules and conservation of momentum, but considering a restitution coefficient that is a function of the material of the colliding objects, modeling dissipating phenomena in the collision. Cushions are modelled with suitable height and contact radius, in order to predict the correct collision behavior. All the dynamics is implemented through the Novodex dynamic simulation engine.

In Figure 5.13 and Figure 5.14 the possibilities offered by the application are shown, like real time collision detection capability and dynamics with modeling of rolling of balls, and possiblity of applying spinning effects when hitting the balls, by varying the point of application of force $\mathbf{p}$.

5.4.2 Virtual Restoration

The Virtual Restoration application was developed as part of the VICOM project, with the objective of providing a haptic enabled system for the collaborative restoration of pictures. In this application HapticWeb is used with the scenegraph form and integrated in a higher level library that allows the mixed use of mouse, local and remote haptic interfaces, in a device independent graphical user interface. Haptics is used also for enhancing the feedback of exploration of the image with simulated roughness.

5.5 Discussion

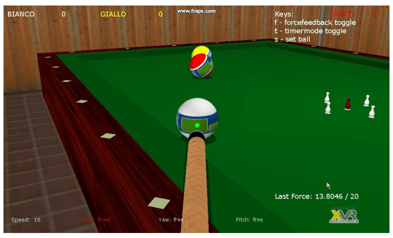
In this chapter we have presented HapticWeb, a tool for the creation of haptic application on the Web. The description of the features and the architecture has been completed by an evaluation of the haptic loop performance under different conditions. We hope that this tool and its application will increase the development and the experimentation on haptics, and the creation of new kinds of applications. Additional information about HapticWeb, the documentation and the code for developing applications are available on the project's website `http://www.hapticweb.org/`.

The main limitations of the HapticWeb framework are the limit in the type of haptic rendering modes provided, effects and meshes, and the complexity in the development, although reduced respect the traditional C++ applications. The integration of the scripting system with patch based visual programming would improve the effectiveness of HapticWeb.

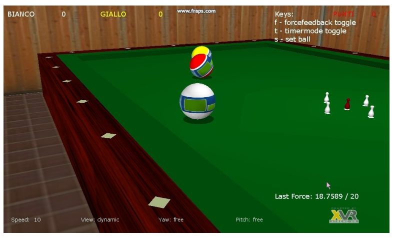
Future plans for HapticWeb cover improvements of the object model for higher level management of the scene, support for more haptic materials as textures, and a complete collaborative layer for the interaction between remote environments. Another more general improvement for HapticWeb is the porting of the XVR system to Java, an effort that involves the creation of a new compiler with support of types and that generates efficient Java code from S3D code. The benefits of this porting there are the support for multiple platforms and the higher efficiency provided by the Java Virtual Machine, two elements that could increase the spreading of this framework.

Acknowledgements

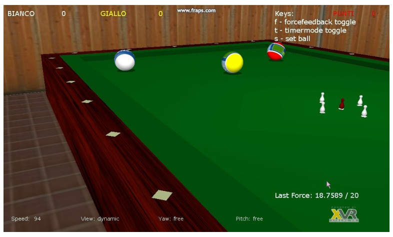
I would like to thanks the development team of the CHAI3D library for their effort in the creation of an Open Source library for haptics and also for their help in the understanding of the library and the resolution of issues. At the same time HapticWeb would be not possible without the work done by the XVR team, in particular Sandro Bacinelli with his patience of while listening for new solutions and the his support during the development.



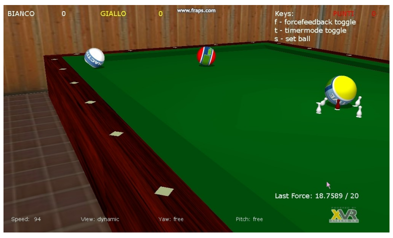
(a) The user manipulates the billiard cue



(b) Soon after the queue has hit the ball that is travelling toward the other balls

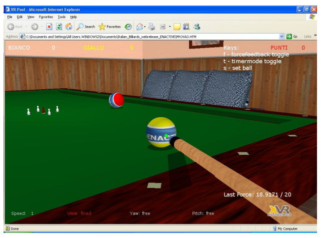


(c) The ball has collided with the other balls

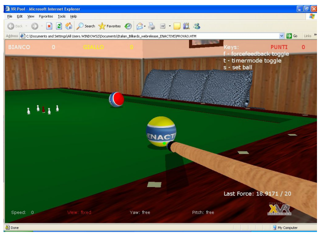


(d) The ball hits cushions and birilli

Figure 5.13: A sequence of snapshots of the pooling demo application



(a) Central hit



(b) Underspinning hit of the ball

Figure 5.14: Possibility of adding spinning effects while hitting the ball

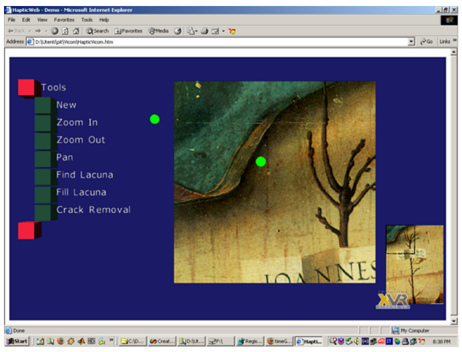


Figure 5.15: Virtual Restoration application, working with the two arms of the GRAB device

Chapter 6

Summary

The research activity presented in previous chapters presented an overall vision of Haptic Rendering from the low level algorithms to high level application design. The focus of the research have been a balance between the management of multirate issues in Haptic Rendering and the perceptual aspects of Haptics.

The future directions of research are aimed at two different aspects. The first is based on an improved understanding of the concept of haptic trajectories not only for imporoving benchmarks but also for the recording, storage and transmission of haptic gestures. The second aspect focuses instead on the aspect of framework for Haptic application based on information, in which a generic information set is transformed into a multimodal representation enhanced with haptic feedback.

Appendix A

GNU Free Documentation License

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **"Document"**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as **"you"**. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A **"Modified Version"** of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A **"Secondary Section"** is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **"Invariant Sections"** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **"Cover Texts"** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **"Transparent"** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called **"Opaque"**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **"Title Page"** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section **"Entitled XYZ"** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **"Acknowledgements"**, **"Dedications"**, **"Endorsements"**, or **"History"**.) To **"Preserve the Title"** of such a section when

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

ADDENDUM: How to use this License for your documents

Bibliography

[1] Otaduy M. A. *6-DoF Haptic Rendering Using Contact Levels of Detail and Haptic Textures.* PhD thesis, University of North Carolina at Chapel Hill, Department of Computer Science, 2004.

[2] M. C. Lin A. Gregory A. Mascarenhas, S. Ehmann and D. Manocha. 6-dof haptic display of polygonal models. In *Proceedings of IEEE Visualization Conference*, 2000.

[3] SenseGraphics AB. H3D API `http://www.h3d.org`.

[4] Y. Adachi, T. Kumano, and K. Ogino. Intermediate representation for stiff virtual objects. In *IEEE Virtual Reality Annual Intl. Symposium*, pages 203–210, Research Triangle Park, N. Carolina, March 1995.

[5] Carlo Avizzano, Teresa Gutierrez, Sara Casado, Blaithin Gallagher, Mark Magennis, John Wood, Keith Gladstone, Helen Graupp, Jose A. Munoz, Elena Francisca Cano Arias, and Fiona Slevin. Grab: Computer graphics access for blind people through a haptic and audio virtual environment. In *Proceedings of Eurohaptics 2003*, July 2003.

[6] David Baraff. Analytical methods for dynamic simulation of non-penetrating rigid bodies. *Computer Graphics*, 23(3):223–232, 1989.

[7] David Baraff. Curved surfaces and coherence for non-penetrating rigid body simulation. *Computer Graphics*, 24(4):19–28, 1990.

[8] David Baraff. Fast contact force computation for nonpenetrating rigid bodies. *Computer Graphics*, 28(Annual Conference Series):23–34, 1994.

[9] F. Barbagli, R. DeVengenzo, and K. Salisbury. Dual-handed virtual grasping. In *2003 IEEE International Conference on Robotics and Automation, ICRA 2003*, volume 1, pages 1259–1263, Taipei, Taiwan, 2003.

[10] F. Barbagli, A. Frisoli, K. Salisbury, and M. Bergamasco. Simulating human fingers: a soft finger proxy model and algorithm. In *Proceedings of the Haptic Symposium*, volume 1, pages 9–17, Chicago, Illinois, March 2004.

[11] Jim Barnett. Multimodal architecture and interfaces,working draft. Technical report, W3C, April 2005.

[12] A.K. Bejczy and J.K. Salisbury. Controlling remote manipulators through kinesthetic coupling. *Computers in Mechanical Engineering*, 2(1):48–60, 1983.

[13] M. Bergamasco, B. Allotta, L. Bosio, L. Ferretti, G. Parrini, G.M. Prisco, F. Salsedo, and G. Sartini. An arm exoskeleton system for teleoperation and virtual environments applications. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1449–1454, 1994.

[14] Massimo Bergamasco, Emanuele Ruffaldi, and Carlo Alberto Avizzano. Enactive and internet applications: A first prototype of enactive application that interoperates haptic devices and the world wide web, 2006.

[15] P. Berkelman, R. Hollis, and D. Baraff. Interaction with a realtime dynamic environment simulation using a magnetic levitation haptic interface device. In *IEEE International Conference on Robotics and Automation*, 1999.

[16] D. Borro, J. Savall, A. Amundarain, JJ Gil, A. Garcia-Alonso, and L. Matey. A large haptic device for aircraft engine maintainability. *Computer Graphics and Applications, IEEE*, 24(6):70–74, 2004.

[17] William J. Bouma and George Vaněček, Jr. Modeling contacts in a physically based simulation. In *SMA '93: Proceedings of the Second Symposium on Solid Modeling and Applications*, pages 409–418, 1993.

[18] D.L. Brock. Enhancing the dexterity of a robot hand using controlled slip. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 249–251, 1988.

[19] J.M. Brown and J.E. Colgate. Factors affecting the z-width of a haptic display. In *Proc. of the IEEE ICRA conference*, 1994.

[20] S. Cameron. Collision detection by four–dimensional intersectin testing. *IEEE Transaction on Robotics and Automation*, 6(3):291–302, 1990.

[21] S. Cameron. Enhancing GJK: computing minimum and penetration distances between convex polyhedra. In *Int. Conf. Robotics & Automation*, April 1997.

[22] M. Carrozzino, F.Tecchia, S.Bacinelli, and M.Bergamasco. Lowering the development time of multimodal interactive application: The real-life experience of the XVR project. In *ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, 2005.

[23] Marcello Carrozzino. *Efficient authoring and management of complex virtual environments.* PhD thesis, Scuola Superiore S.Anna, Laboratorio PERCRO, 2006.

[24] B. Chang and JE. Colgate. Real-time impulse-based simulation of rigid body systems for haptic display. In *Proc. Symp. on Interactive 3D Graphics*, 1997.

[25] Hui Chen and Hanqiu Sun. Body-based haptic interaction model for touch-enabled virtual environments. *Presence: Teleoperators and Virtual Environments*, 15(2):186–203, 2006.

[26] S. Choi and H. Z. Tan. An analysis of perceptual instability during haptic texture rendering. In *10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS'02)*, 2002.

[27] S. Choi and H. Z. Tan. Effect of update rate on perceived instability of virtual haptic texture. In *Intl. Conference on Intelligent Robots and Systems*, 2004.

[28] S. Choi and H. Z. Tan. Perceived instability of virtual haptic texture. i. experimental studies. In *Presence: Teleoperators & Virtual Environments*, 2004.

[29] Jonathan D. Cohen, Ming C. Lin, Dinesh Manocha, and Madhav K. Ponamgi. I-COLLIDE: An interactive and exact collision detection system for large-scale environments. In *Symposium on Interactive 3D Graphics*, pages 189–196, 218, 1995.

[30] E. Colgate, P. Grafing, M. Stanley, and G. Schenkel. Implementation of stiff virtual walls in force-reflecting interfaces. In *Proceedings of VRAIS*, pages 202–208, Seattle, WA, September 1993.

[31] J. Colgate, M. Stanley, and J. Brown. Issues in the haptic display of tool use. In *Int'l Conf. on Intelligent Robots and Systems*, Pittsburgh, PA, August 1995.

[32] J. Edward Colgate and Gerd G. Schenkel. Passivity of a class of sampled-data systems: Application to haptic interfaces. *Journal of Robotic Systems*, 14:37–47, 1997.

[33] D. Constantinescu, S.E. Salcudean, and E.A. Croft. Impulsive forces for haptic rendering of rigid contacts. In *Proceedings of the 35th International Symposium on Robotics*, 2004.

[34] D. Constantinescu, S.E. Salcudean, and E.A. Croft. Haptic rendering of rigid contacts using impulsive and penalty forces. *IEEE Transactions on Robotics*, 21(3):309–323, 2005.

[35] Daniela Constantinescu, Septimiu E. Salcudean, and Elizabeth A. Croft. Local model of interaction for haptic manipulation of rigid virtual worlds. *Int. J. Rob. Res.*, 24(10):789–804, 2005.

[36] Francois Conti, Federico Barbagli, Dan Morris, and Christopher Sewell. CHAI, an open-source library for the rapid development of haptic scenes. In *Hands-On-Demo at the World Haptics Conference*, 2005.

[37] Morris D., Sewell C., Blevins N., Barbagli F., and Salisbury K. A collaborative environment for the simulation of temporal bone surgery. In *Proceedings of MICCAI (Medical Image Computing and Computer-Aided Intervention) VII, Rennes, France, September 26-30*, 2004.

[38] M. de Pascale, G. de Pascale, D. Prattichizzo, and F. Barbagli. The Haptik Library: a Component based Architecture for Haptic Devices Access. In *Proceedings of Eurohaptic*, 2004.

[39] Maurizio de Pascale, Gabriele Sarcuni, and Domenico Prattichizzo. Real-time soft-finger grasping of physically based quasi-rigid objects. In *WHC*, pages 545–546, 2005.

[40] A. Dettori, C.A.Avizzano, S. Marcheschi, M.Angerilli, M. Bergamasco, C.Loscos, and A.Guerraz. Art touch with create haptic interface. In *ICAR 2003, 11th International Conference on Advanced Robotics*, 2003.

[41] David P. Dobkin, John Hershberger, David G. Kirkpatrick, and Subhash Suri. Computing the intersection-depth of polyhedra. *Algorithmica*, 9(6):518–533, 1993.

[42] P. Dupont, V. Hayward, B. Armstrong, and F. Altpeter. Single state elasto-plastic friction models. *IEEE Transactions on Automatic Control*, 47(5):787–792, 2002.

[43] S. Ehmann and M. Lin. Accelerated proximity queries between convex polyhedra by multi-level voronoi marching. Technical report, Computer Science Department, University of North Carolina at Chapel Hill, 2000., 2000.

[44] D. Feygin, M. Keehner, and F. Tendick. Haptic guidance: Experimental evaluation of a haptic training method for a perceptual motor skill. In *Proceedings of the 10th IEEE Haptics Symposium*, 2002.

[45] Jonathan Fiene, Katherine J. Kuchenbecker, and Günter Niemeyer. Event-based haptics with grip force compensation. In *Proc. Haptic Symposium*, March 2006.

[46] Jansson G., Faenger J., Konig H., and Billberger K. Visually impaired persons' use of PHANToM for information about texture and 3D form of virtual objects. In *Proceedings of the Third PHANToM User's Group, PUG98*, 1998.

[47] RS Johansson G. Westling. Factors influencing the force control during precision grip. *Exp Brain Res*, 53:277–284, 1984.

[48] J. J. Gibson. *The senses considered as perceptual systems.* Houghton Mifflin, 1966.

[49] E G Gilbert, D W Johnson, and S S Keerthi. A fast procedure for computing the distance between complex objects in three dimensional space. *IEEE J. Robotics and Automation*, 4(2):193–203, 1988.

[50] B. Gillespie, M. Cutkosky, and S. UserSpeci. Rendering of the virtual wall. In *Proceedings of the ASME International Mechanical Engineering Conference and Exposition*, volume 58, pages 397–406, Atlanta, GA, November 1996. ASME.

[51] R.C. Goertz. Fundamentals of general-purpose remote manipulators. *Nucleonics*, 10:36–42, 1952.

[52] Antony W. Goodwin, Per Jenmalm, and Roland S. Johansson. Control of grip force when tilting objects: Effect of curvature of grasped surfaces and applied tangential torque. *The Journal of Neuroscience*, 18(24):10724–10734, 1998.

[53] S. Gottschalk, M. C. Lin, and D. Manocha. OBBTree: A hierarchical structure for rapid interference detection. *Computer Graphics*, 30(Annual Conference Series):171–180, 1996.

[54] A. E. Green and J. E. Adkins. *Large elastic deformatoins.* Oxford at Clarendon Press, 1960.

[55] U.O. Gretarsdottir, F. Barbagli, and J.K. Salisbury. Phantom-X. In *Proceedings of EuroHaptics*, 2003.

[56] Eran Guendelman, Robert Bridson, and Ronald Fedkiw. Nonconvex rigid bodies with stacking. *ACM Trans. Graph.*, 22(3):871–878, 2003.

[57] A. Guerraz, C. Loscos, and H.R. Widenfeld. How to use physical parameters coming from the haptic device itself to enhance the evaluation of haptic benefits in user interface? In *Proceedings of EuroHaptics*, 2003.

[58] Leonidas J. Guibas, David Hsu, and Li Zhang. H-walk : Hierarchical distance computation for moving convex bodies. In *Symposium on Computational Geometry*, pages 265–273, 1999.

[59] R. J. Gulati and M. A. Srinivasan. Human fingerpad under indentation i: static and dynamic force response. In *Proc. of Bioengineering Conference (BED-Vol. 29)*, pages 261–262, 1995.

[60] H. Cenk Guler, N. Berme, and S. R. Simon. A viscoelastic sphere model for the representation of plantar soft tissue during simulation. *Journal of Biomechanics*, 31:847–53, 1998.

[61] Shoichi Hasegawa, Nobuaki Fujii, Yasuharu Koike, and Makoto Sato. Real-time rigid body simulation based on volumetric penalty method. In *HAPTICS*, pages 326–332. IEEE Computer Society, 2003.

[62] V. Hayward and O. Astley. Performance measures for haptic interfaces. In Eds. Giralt G., Hirzinger G., editor, *1996 Robotics Research: The 7th International Symposium*, pages 195–207. Springer Verlag, 1996.

[63] H. Hertz. On the contact of elastic solids. *J. Reine und angewandte Mathematik*, 92:156–171, 1882. (for english translation see *Miscellaneous Papers by H. Hertz*, Eds Jones and Schott, London: Macmillan 1896).

[64] N. Hogan. Conkrolling impedance at man-machine interface. In *Proceedings of ICRA*, 1989.

[65] Philip M. Hubbard. Space-time bounds for collision detection. Technical Report CS-93-04, Brown University, 1993.

[66] Philip M. Hubbard. Approximating polyhedra with spheres for time-critical collision detection. *ACM Transactions on Graphics*, 15(3):179–210, 1996.

[67] Jesse D. Hwang, Michael D. Williams, and Günter Niemeyer. Toward event-based haptics: Rendering contact using open-loop force pulses. In *HAPTICS*, pages 24–31, 2004.

[68] Ian Iacobs and Norman Walsh. Architecture of the World Wide Web, Volume One, Reccomendation. Technical report, W3C, December 2004.

[69] Immersion Corporation - the CyberGrasp system - `http://www.immersion.com`.

[70] B. Itkowitz, J. Handley, and W. Zhu. The OpenHaptics toolkit: a library for adding 3D Touch, navigation and haptics to graphics applications. In *Proceedings of the 1st World Haptic Conference*, pages 590–591, 2005.

[71] B. Hannaford J. H. Ryu. Time domain passivity control of haptic interfaces. *IEEE Transactions on Robotics and Automation*, 2001.

[72] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264Ű323, 1999.

[73] V. I. Johannes, M. A. Green, and C. A. Brockley. The role of the rate of application of the tangential force in determining the statwc friction coefficient. *Wear*, 24:381–85, 1973.

[74] David E. Johnson and Elaine Cohen. Spatialized normal come hierarchies. In *SI3D*, pages 129–134, 2001.

[75] David E. Johnson and Peter Willemsen. Accelerated haptic rendering of polygonal models through local descent. In *HAPTICS*, pages 18–23, 2004.

[76] Hartmut Keller, Horst Stolz, and Andreas Ziegler. Virtual mechanics : Simulation and animation of rigid body systems. Technical Report TR-1993-08, University of Stuttgart, IPVR, Computer Vision Group, 1993.

[77] Young J. Kim, Miguel A. Otaduy, Ming C. Lin, and Dinesh Manocha. Six-degree-of-freedom haptic rendering using incremental and localized computations. *Presence: Teleoper. Virtual Environ.*, 12(3):277–295, 2003.

[78] Sukhatme G. S. Kim L., Kyrikou A. An implicit-based haptic rendering technique. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2943–2948, 2002.

[79] H. Kinoshita, L. Backstrom, J. R. Flanagan, and R. S. Johansson. Tangential torque effects on the control of grip forces when holding objects with a precision grip. *J. Neurophysiol.*, 1(78):1619–1630, 1998.

[80] T. Kirkpatrick and S. Douglas. Application-based evaluation of haptic interfaces. In *Proceedings of IEEE Virtual Reality conference, VR2002*, Orlando, FL, may 2002.

[81] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan. Efficient collision detection using bounding volume hierarchies of k-DOPs. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):21–36, /1998.

[82] Katherine J. Kuchenbecker, Jonathan Fiene, and Günter Niemeyer. Event-based haptics and acceleration matching: Portraying and assessing the realism of contact. In *Proc. World Haptics Conference*, March 2005.

[83] D. Lawrence, L. Pao, A. Dougherty, M. Salada, and Y. Pavlou. Rate-hardness: a newperformance metric for haptic interfaces. *IEEE Transactions on Robotics and Automation*, 16(4):357–371, 2000.

[84] J. C. Liao and M. A. Srinivasan. Experimental investigation of frictional properties of the human fingerpad. Technical report, MIT Touch Lab, Cambridge Mass, 1999.

[85] Ming Lin and John Canny. A fast algorithm for incremental distance calculation. In *IEEE Int. Conf. Robotics and Automation*, 1994.

[86] K. London, S. Moore, P. Mucci, K. Seymour, and R. Luczak. The PAPI cross-platform interface to hardware performance counters. In *Department of Defense Users' Group Conference Proceedings*, 2001.

[87] Karljohan Lundin, Björn Gudmundsson, and Anders Ynnerman. General proxy-based haptics for volume visualization. In *WHC*, pages 557–560, 2005.

[88] Bergamasco M., C. A. Avizzano, G. Di Pietro, and F. Barbagli. The museum of Pureform: System architecture. In *Proceedings Of 10th IEEE International Workshop on Robot Human Interaction 2001*, pages 112–117, Paris, France, 2001.

[89] W. A. McNeely M. Wan. Quasi-static approximation for 6 degrees-of-freedom haptic rendering. *Proc. of IEEE Visualization*, pages 257–262, 2003.

[90] M. Mahvash and V. Hayward. High fidelity passive force reflecting virtual environments. *IEEE Transactions on Robotics*, 21(1):38–46, 2005.

[91] William R. Mark, Scott C. Randolph, Mark Finch, James M. Van Verth, and II Russell M. Taylor. Adding force feedback to graphics systems: issues and solutions. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 447–452, New York, NY, USA, 1996. ACM Press.

[92] T. H. Massie and J.K. Salisbury. The phantom haptic interface: A device for probing virtual objects. In *Proceedings of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 1994.

[93] Michael McKenna and David Zeltzer. Dynamic simulation of autonomous legged locomotion. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 29–38, New York, NY, USA, 1990. ACM Press.

[94] Troy J. McNeeily A., Puterbaugh K. Voxel-based 6-dof haptic rendering improvements. *Haptics-e*, 3,7, 2005.

[95] N. Melder and W. S. Harwin. Extending the friction cone algorithm for arbitrary polygon based haptic objects. In *HAPTICS*, pages 234–241, 2004.

[96] Ming C. Lin Miguel A. Otaduy. Stable and responsive six-degree-of-freedom haptic manipulation using implicit integration. In *Proceedings of IEEE Worldhaptics05*, 2005.

[97] Brian Mirtich. V-clip: fast and robust polyhedral collision detection. *ACM Trans. Graph.*, 17(3):177–208, 1998.

[98] Brian Mirtich and John F. Canny. Impulse-based dynamic simulation. In *Workshop on Algorithmic Foundations of Robotics*, page 24, 1994.

[99] McLaughlin M.L., Sukhatme G., Shahabi C., Medioni G., and Jaskowiak J. The Haptic Museum. In *Proceedings of the EVA 2000 Conference on Electronic Imaging and the Visual Arts, Florence*, 2000.

[100] Taylor M.M., S.J. Lederman, and R.H. Gibson. Tactual perception of texture. *Handbook of Perception*, pages 251–272, 1973.

[101] Matthew Moore and Jane Wilhelms. Collision detection and response for computer animationr3. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 289–298, New York, NY, USA, 1988. ACM Press.

[102] David E. Muller and Franco P. Preparata. Finding the intersection of two convex polyhedra. *Theor. Comput. Sci.*, 7:217–236, 1978.

[103] A. Nahvi and J.M. Hollerbach. Display of friction in virtual environments based on human finger pad characteristics. In *Proceedings of ASME Dynamic Systems and Control Division*, volume DSC-Vol. 64, pages 179–184, November 1998.

[104] D. Nelson, D. Johnson, and E. Cohen. Haptic rendering of surface-to-surface sculpted model interaction. In *8th Annual Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems, Nashville, TN.*, 1999.

[105] A.M. Okamura, N. Smaby, and M.R. Cutkosky. An overview of dexterous manipulation. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation, Symposium on Dexterous Manipulation*, pages 255–262, 2000.

[106] Marcia Kilchenman O'Malley and Shannon Hughes. Simplified authoring of 3D haptic content for the World Wide Web. In *HAPTICS*, pages 428–429, 2003.

[107] M. Otaduy and M. Lin. Sensation preserving simplification for haptic rendering. In *Proceedings of ACM SIGGRAPH*, 2003.

[108] D. K. Pai, J. Lang, J. E. Lloyd, and R. J. Woodham. Acme, a telerobotic active measurement facility. In *Proceedings of the Sixth International Symposium on Experimental Robotics*, March 1999.

[109] D. K. Pai, K. van den Doel, D. L. James, J. E J. Lang, Lloyd, J. L. Richmond, and S. H. Yau. Scanning physical interaction behavior of 3d objects. In *Computer Graphics (ACM SIGGRAPH 2001 Conference Proceedings)*, August 2001.

[110] D.T.V. Pawluk and R.D. Howe. Dynamic contact of the human fingerpad against a flat surface. *Journal of Biomechanical Engineering*, 121:605–611, 1999.

[111] D.T.V. Pawluk and R.D. Howe. Dynamic lumped element response of the human fingerpad. *Journal of Biomechanical Engineering*, 121:178–183, 1999.

[112] M. K. Ponamgi, D. Manocha, and M. C. Lin. Incremental algorithms for collision detection between polygonal models:. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):51–64, /1997.

[113] Anthony Prior and Karen G. Haines. The use of a proximity agent in a collaborative virtual environment with 6 degrees-of-freedom voxel-based haptic rendering. In *WHC*, pages 631–632, 2005.

[114] C. Raymaekers, J. De Boeck, and K. Coninx. An empirical approach for the evaluation of haptic algorithms. In *Proceedings of IEEE World Haptics*, 2005.

[115] M. Rossi, K. Tuer, and D. Wang. A new design paradigm for the rapid development of haptic and telehaptic applications. *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pages 1246–1250, 2005.

[116] Emanuele Ruffaldi. Scripting for setup of haptic experiments. Technical Report Diploma di Licenza, PERCRO, Scuola Superiore S.Anna, 2003.

[117] Emanuele Ruffaldi, Sandro Bacinelli, Marcello Carrozzino, Franco Tecchia, and Massimo Bergamasco. Dual scripting in a Virtual Reality engine. embedding Python in XVR. In *Proceedings of EuroPython*, 2006.

[118] Emanuele Ruffaldi and Chiara Evangelista. Populating virtual environments using semantic web. In *Proceedings of Semantic Web Application and Perspectives*, 2004.

[119] Emanuele Ruffaldi, Dan Morris, Timothy Edmunds, Federico Barbagli, and Dinesh K.Pai. Standardized evaluation of haptic rendering systems. In *Proceedings of the Haptic Symposium*, 2006.

[120] D. Ruspini and O. Khatib. Dynamic models for haptic rendering systems. In *Advances in Robot Kinematics: ARK98, Strobl/Salzburg, Austria*, pages 523–532, 1998.

[121] D. Ruspini and O. Khatib. A framework for multi-contact multi-body dynamic simulation and haptic display. In *Proc. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems. Vincent Hayward and Dingrong Yi*, 2000.

[122] Diego C. Ruspini, Krasimir Kolarov, and Oussama Khatib. The haptic display of complex graphical environments. *Computer Graphics*, 31(Annual Conference Series):345–352, 1997.

[123] S. Salcudean and T. Vlaar. the emulation of stiff walls and static friction with a magnetically levitated input/output device. *ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems, Dynamics Systems and Control*, 119:127–132, March 1997.

[124] K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles. Haptic rendering: programming touch interaction with virtual objects. *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 123–130, 1995.

[125] K. Salisbury and Mason. *Robot Hands and the Mechanics of Manipulation.* MIT press, 1985.

[126] F. Salsedo, M. Fontana, F. Tarri, E. Ruffaldi, M. Bergamasco, U. Bonanni N. Magnenat-Thalmann, P. Volino, A. Brady, I. Summers, J. Qu, D.Allerkamp, G. Bottcher, F-E Wolter, M.Makinen, and H.Meinander. Architectural design of the haptex system. In *HAPTEXŠ05 Workshop on Haptic and Tactile Perception of Deformable Objects*, December 2005.

[127] Reachin (SE). Reachin API http://www.reachin.se.

[128] Inc. SensAble Technologies. OpenHaptics toolkit http://www.sensable.com.

[129] E.R. Serina, E. Mockensturm, C.D. Mote, and D. Rempel. A structural model of the forced compression of the fingertip pulp. *Journal of Biomechanics*, 31:639–646, 1998.

[130] E.R. Serina, C.D. Mote, and D. Rempel. Force response of the fingertip pulp to repeated compression: Effects of loading rate, loading angle, antropometry. *Journal of Biomechanics*, 30:1035–1040, 1997.

[131] Scott S. Snibbe, Karon E. MacLean, Rob Shaw, Jayne Roderick, William Verplank, and Mark Scheeff. Haptic techniques for media control. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 199–208, 2001.

[132] John M. Snyder. An interactive tool for placing curved surfaces without interpenetration. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 209–218, New York, NY, USA, 1995. ACM Press.

[133] John M. Snyder, Adam R. Woodbury, Kurt Fleischer, Bena Currin, and Alan H. Barr. Interval methods for multi-point collisions between time-dependent curved surfaces. *Computer Graphics*, 27(Annual Conference Series):321–334, 1993.

[134] M.A. Srinivasan. Surface deflection of primate fingertip under line load. *Biomechanicals*, 22(4):343–349, 1989.

[135] M.A. Srinivasan and K. Dandekar. An investigation of the mechanics of tactile sense using two dimensional models of the primate fingertip. *Journal of Biomechanical Engineering*, 118:48–55, 1996.

[136] Mandayam A. Srinivasan and Cagatay Basdogan. Haptics in virtual environments: taxonomy, research status, and challenges. *Computer Graphics*, 21(4):393–404, July–August 1997.

[137] H.Z. Tan. Identification of sphere size using the PHANToM: Towards a set of building blocks for rendering haptic environments. In *Proceedings of the ASME Annual Meeting*, volume 61, Nov 1997.

[138] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 205–214, New York, NY, USA, 1987. ACM Press.

[139] Bernard Traversat, Mohamed Abdelaziz, Dave Doolin, Mike Duigou, Jean-Christophe Hugly, and Eric Pouyoul. Project JXTA-C: Enabling a web of things. In *HICSS*, page 282, 2003.

[140] P. Astley V. Hayward, P. Gregorio and S O. Greenish. Freedom-7: A high fidelity seven axis haptic device with application to surgical training. In *Experimental Robotics V, Lecture Notes in Control and Information Science 232*, pages 445–456, 1998.

[141] Gino van den Bergen. Efficient collision detection of complex deformable models using aabb trees. *J. Graph. Tools*, 2(4):1–13, 1997.

[142] Gino van den Bergen. A fast and robust GJK implementation for collision detection of convex objects. *Journal of Graphics Tools: JGT*, 4(2):7–25, 1999.

[143] The Virtual Reality Modeling Language (VRML 97), International Specification ISO/IEC 14772-1, Dicembre 1997.

[144] K. Puterbaugh W. McNeely and J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *Proceedings of ACMSIGGRAPH*, page 401Ű408, 1999.

[145] G. Westling and R. S. Johansson. Factors influencing the force control during precision grip. *Experimental Brain Research*, 53:277–284, 1998.

[146] Gabriel Zachmann. Minimal hierarchical collision detection. In *Proc. ACM Symposium on Virtual Reality Software and Technology (VRST)*, pages 121–128, Hong Kong, China, November11–13 2002.

[147] C. Zilles and J. Salisbury. A constraintbased god-object method for haptic display. In *Proc. IEE/RSJ International Conference on Intelligent Robots and Systems, Human Robot Interaction, and Cooperative Robots*, volume 3, pages 146–151, 1995.

[148] C. B. Zilles and J. K. Salisbury. A constraint-based god-object method for haptic display. In *Proc. of IROS*, 1995.