High Fidelity Haptic Rendering

High Fidelity Haptic Rendering
Miguel A. Otaduy and Ming C. Lin
www.morganclaypool.com

ISBN: 1598291149 paperback
ISBN: 9781598291148 paperback

ISBN: 1598291157 ebook
ISBN: 9781598291155 ebook

DOI: 10.2200/S00045ED1V01Y200609CGR002

A Publication in the Morgan & Claypool Publishers' series
SYNTHESIS LECTURES IN COMPUTER GRAPHICS AND ANIMATION #2

Lecture #2
Series Editor: Brian A. Barsky, University of California, Berkeley

10 9 8 7 6 5 4 3 2 1

High Fidelity Haptic Rendering

Miguel A. Otaduy
ETH Zurich

Ming C. Lin
University of North Carolina, Chapel Hill

SYNTHESIS LECTURES IN COMPUTER GRAPHICS AND ANIMATION #2

MORGAN & CLAYPOOL PUBLISHERS

ABSTRACT

The human haptic system, among all senses, provides unique and bidirectional communication between humans and their physical environment. Yet, to date most human-computer interactive systems have focused primarily on the graphical rendering of visual information and, to a lesser extent, on the display of auditory information. Extending the frontier of visual computing, haptic interfaces, or force feedback devices, have the potential to increase the quality of human-computer interaction by accommodating the sense of touch. They provide an attractive augmentation to visual display and enhance the level of understanding of complex data sets. They have been effectively used for a number of applications including molecular docking, manipulation of nano-materials, surgical training, virtual prototyping and digital sculpting. Compared with visual and auditory display, haptic rendering has extremely demanding computational requirements. In order to maintain a stable system while displaying smooth and realistic forces and torques, high haptic update rates in the range of 500–1000 Hz or more are typically used. Haptics presents many new challenges to researchers and developers in computer graphics and interactive techniques. Some of the critical issues include the development of novel data structures to encode shape and material properties, as well as new techniques for geometry processing, data analysis, physical modeling, and haptic visualization.

This synthesis examines some of the latest developments on haptic rendering, while looking forward to exciting future research in this area. It presents novel haptic rendering algorithms that take advantage of the human haptic sensory modality. Specifically it discusses different rendering techniques for various geometric representations (e.g. point-based, polygonal, multiresolution, distance fields, etc), as well as textured surfaces.

It also shows how psychophysics of touch can provide the foundational design guidelines for developing perceptually driven force models and concludes with possible applications and issues to consider in future algorithmic design, validating rendering techniques, and evaluating haptic interfaces.

KEYWORDS

Force Display, Collision Detection, Dynamic Simulation, Multi-resolution Representation, Level-of-Detail Techniques, Haptic Texture Rendering, Psychophysics of Touch, Control and Stability

Contents

CHAPTER 1

Fundamentals of Haptic Rendering

The ability to interact with synthetic entities as if they were real has been the ultimate quest of virtual reality (VR) researchers for decades. Recent advances in virtual environments allow us to *see* virtual objects and avatars, to *hear* them, to *move* them, and to *touch* them. The direct physical interaction with computer-generated objects enabled by haptic interfaces provides a useful and intuitive augmentation to visual display and the opportunity to enhance the level of understanding of, and interactivity with, complex data sets. Haptic technologies are already used effectively in a number of novel applications including molecular docking, manipulation of nanomaterials, surgical training, virtual prototyping, and digital sculpting. It has also been shown that the ability to touch virtual objects increases the sense of presence in virtual environments [Ins01].

However, in comparison with visual and auditory display, haptic rendering has extremely demanding computational requirements. In order to maintain a stable system while displaying smooth and realistic forces and torques, haptic update rates of 1 kHz or more are typically used. Haptics presents new challenges in the development of novel data structures to render objects and abstract concepts, to encode shape and material properties, as well as new techniques for data processing and analysis, physical modeling, and visualization.

This synthesis describes some of the latest developments on haptic rendering and applications. It provides an introductory view to the field. This synthesis will first briefly review the basic concepts on force display, such as psychophysics of haptics, control, and stability. The central focus will be on basic methodologies for haptic rendering of interaction between *two* three-dimensional objects. Since collision detection and dynamic simulation are integral to fast computation of interaction forces between two 3D objects, a majority of this synthesis will be devoted to these topics, especially between rigid bodies and textured objects. We refer the readers to [Bur96] for details on force feedback hardware and tactile interface design.

1.1 INTRODUCTION

Before covering in depth the haptic display of interaction between virtual objects, in this chapter we introduce the concept of haptic rendering, related fundamental research in psychophysics and control theory, and basic techniques for haptic rendering through point-based interaction.

We first start by defining some basic terminology, discussing the evolution of the research in haptic rendering, and introducing some interesting applications.

1.1.1 Definitions

The term *haptic* (from the Greek *haptesthai*, meaning "to touch") is the adjective used to describe something relating to or based on the sense of touch. Haptic is to touching as visual is to seeing and as auditory is to hearing [FFM+04].

As described by Klatzky and Lederman [KL03], touch is one of the key sensory channels and it can be divided into cutaneous, kinesthetic, and haptic systems, based on the underlying neural inputs. The cutaneous system employs receptors embedded in the skin, while the kinesthetic system employs receptors located in muscles, tendons, and joints. The haptic sensory system employs both cutaneous and kinesthetic receptors, but it differs in the sense that it is associated with an active procedure. Touch becomes active when the sensory inputs are combined with controlled body motion. For example, cutaneous touch becomes active when we explore a surface or grasp an object, while kinesthetic touch becomes active when we manipulate an object and touch other objects with it.

Haptic rendering is defined as the process of computing and generating forces in response to user interactions with virtual objects [SBM+95]. Some of the earlier haptic rendering algorithms consider the approach of touching virtual objects with a single contact point. Such rendering algorithms are typically referred to as three-degree-of-freedom (3-DoF) haptic rendering algorithms, because a point in 3D has only three DoF. Other haptic rendering algorithms deal with the problem of rendering the forces and torques arising from the interaction of two objects, often encountered in our daily routines. This problem is often called 6-DoF haptic rendering, because the object virtually attached to the haptic device has six DoF (position and orientation in 3D), and the haptic feedback comprises 3D force and torque. For example, when we eat with a fork, write with a pen, or open a lock with a key, we are manipulating an object in 3D and we feel its interaction with other objects. This is, in essence, 6-DoF object interaction with force-and-torque feedback. Fig. 1.1 shows an example of a user experiencing 6-DoF haptic display. In this example, as he manipulates an object and touches other objects with it, he perceives cutaneous feedback as the result of grasping and kinesthetic feedback as the result of contact between objects.

1.1.2 From Telerobotics to Haptic Rendering

In 1965, Ivan Sutherland [Sut65] proposed a multimodal display that would incorporate haptic feedback into the interaction with virtual worlds. Before then, haptic feedback had already been used mainly in two applications: flight simulators and master–slave robotic teleoperation. The early teleoperator systems had mechanical linkages between the master and the slave. But, in

FIGURE 1.1: *Example of haptic rendering.* A person manipulates a virtual jaw using a haptic device (shown on the right of the image), and the interaction between jaws is displayed both visually and haptically.

1954, Goertz and Thompson [GT54] developed an electrical servomechanism that received feedback signals from sensors mounted on the slave and applied forces to the master, thus producing haptic feedback.

From there, haptic interfaces evolved in multiple directions, but there were two major breakthroughs. The first breakthrough was the idea of substituting the slave robot by a simulated system, in which forces were computed using physically based simulations. The GROPE project at the University of North Carolina at Chapel Hill [BOYBK90], lasting from 1967 to 1990, was the first one to address the synthesis of force feedback from simulated interactions. In particular, the aim of the project was to perform real-time simulation of 3D molecular-docking forces. The second breakthrough was the advent of computer-based Cartesian control for teleoperator systems [BS80], enabling a separation of the kinematic configurations of the master and the slave. Later, Cartesian control was applied to the manipulation of simulated slave robots [KB91].

Those first haptic systems were able to simulate the interaction of simple virtual objects only. Perhaps the first project to target computation of forces in the interaction with objects with rich geometric information was Minsky's seminal work on the *Sandpaper* project [MOyS+90]. Minsky et al. developed a planar force feedback system that allowed the exploration of textures. A few years after Minsky's work, Zilles and Salisbury presented an algorithm for 3-DoF haptic rendering of polygonal models [ZS95]. Almost in parallel with Zilles and Salisbury's work, Massie and Salisbury [MS94] designed the PHANTOM, a stylus-based haptic interface, which

was later commercialized and has become one of the most commonly used force-feedback devices.

By the late 1990s, research in haptic rendering revived one of the problems that first inspired virtual force feedback: 6-DoF haptic rendering or, in other words, grasping of a virtual object and synthesis of kinesthetic feedback of the interaction between the object and its environment.

Research in the field of haptics over the last 35 years has covered many more areas than what we have summarized here. An excellent general survey of the field of haptics is given in [Bur96] and a nice discussion on current research topics is presented in [MHS02]. In this synthesis, we mainly focus on computational techniques, data structures, and geometric algorithms for haptic rendering.

1.1.3 Haptic Rendering for Virtual Manipulation

Certain professional activities, such as training for high-risk operations or preproduction prototype testing, can benefit greatly from "simulated or virtual reproductions". The fidelity of the simulated reproductions depends, among other factors, on the similarity of the behaviors of real and virtual objects. In the real world, solid objects cannot interpenetrate. Contact forces can be modeled mathematically as constraint forces imposed by penetration constraints. However, unless penetration constraints are explicitly imposed, simulated objects are free to penetrate each other in virtual environments. In fact, one of the most disconcerting experiences in synthetic environments is to pass through virtual objects [IMWB01, SU93]. Virtual environments require the simulation of nonpenetrating rigid body dynamics, and this problem has been extensively explored in the robotics and computer graphics literature [Bar92, Mir96].

It has been shown that being able to touch physical replicas of virtual objects (a technique known as *passive haptics* [Ins01]) increases the sense of presence in virtual environments. This conclusion can probably be generalized to the case of synthetic cutaneous feedback of the interaction with virtual objects. As reported by Brooks et al. [BOYBK90], kinesthetic feedback radically improved situation awareness in virtual 3D molecular docking. Kinesthetic feedback has proved to enhance task performance in applications such as telerobotic object assembly [HS77], virtual object assembly [UNB+02], and virtual molecular docking [OY90]. More specifically, task completion time is shorter with kinesthetic feedback in docking operations but not in prepositioning operations.

To summarize, haptic rendering is noticeably useful in some specific examples of training for high-risk operations or preproduction prototype testing activities that involve intensive object manipulation and frequent interaction with the environment. Such examples include minimally invasive or endoscopic surgery [EHS+97, HGA+98] and virtual prototyping for

assembly and maintainability assessment [MPT99, Che99, And02, WM03]. Force feedback becomes particularly important and useful in situations with limited visual feedback.

1.2 COMPUTATIONAL CHALLENGES

Haptic rendering is, in essence, an interactive experience and its realization is mostly handicapped by two conflicting challenges: the required high update rates and the resulting computational cost. In this section we outline the computational pipeline of haptic rendering and discuss the associated challenges.

1.2.1 Haptic Rendering Pipeline

Haptic rendering comprises two main tasks. One of them is the computation of the position and/or orientation of the virtual tool grasped by the user. The other one is the computation of contact force and torque that are displayed back to the user. The existing methodologies for haptic rendering can be classified into two major categories based on their overall computational pipelines.

In *direct rendering* methods [NJC99, GME+00, KOLM03, JW03, JW04], the position and orientation of the haptic device are applied directly to the grasped virtual tool. Collision detection is performed between the grasped tool and other virtual objects, and collision response is applied to the grasped tool as a function of object separation or penetration depth. The resulting contact force and/or torque are directly displayed back to the user.

In *simulation-based* methods [CC97, Ber99, MPT99, RK00, WM03, OL05, ORC06], user actions produce force and torque that drive the motion of the virtual tool. One possibility is to define the force and torque by setting a virtual viscoelastic coupling [CSB95] between the position and orientation of the haptic device and the position and orientation of the virtual tool. Collision detection and response are performed between the virtual tool and other virtual objects. The coupling force and torque are combined with the collision response in order to compute the position and orientation of the virtual tool. The same coupling force and torque are then sent back to the user.

In Chapter 2, we describe different existing methods for 6-DoF haptic rendering in more detail, and discuss their advantages and disadvantages. Also, as explained in Section 1.4, there are two main classes of haptic devices, impedance and admittance devices, which impose slight variations in the haptic rendering pipelines.

1.2.2 Force Update Rate

The ultimate goal of haptic rendering is to provide force feedback to the user. This goal is achieved by controlling the handle of the haptic device, which is in fact the end-effector of a robotic manipulator. When the user holds the handle, he or she experiences kinesthetic

feedback. The entire haptic rendering system is regarded as a mechanical impedance that sets a transformation between the position and velocity of the handle of the device and the applied force.

The quality of haptic rendering can be measured in terms of the dynamic range of impedances that can be simulated in a stable manner [CB94]. When the user moves the haptic device in free space, the perceived impedance should be very low (i.e., small force) and when the virtual tool touches rigid virtual objects, the perceived impedance should be high (i.e., high stiffness and/or damping of the constraint). The quality of haptic rendering can also be measured in terms of the responsiveness of the simulation [BOYBK90, Ber99]. In free-space motion the virtual tool should respond quickly to the motion of the user. Similarly, when the virtual tool collides with a virtual wall, the user should stop quickly, in response to the motion constraint.

With impedance-type devices, virtual walls are implemented as large stiffness values in the simulation. In haptic rendering, the user is part of a closed-loop sampled dynamic system [CS94], along with the device and the virtual environment, and the existence of sampling and latency phenomena can induce unstable behavior under large stiffness values. System instability is directly perceived by the user in the form of disturbing oscillations. A key factor for achieving a high dynamic range of impedances (i.e., stiff virtual walls) while ensuring stable rendering is the computation of feedback forces at a high update rate [CS94, CB94]. Brooks et al. [BOYBK90] reported that, in the rendering of textured surfaces, users were able to perceive performance differences at force update rates between 500 Hz and 1 kHz.

A more detailed description of the stability issues involved in the synthesis of force feedback and a description of related work are given in Section 1.4. Although here we have focused on impedance-type haptic devices, similar conclusions can be drawn for admittance-type devices (see [AH98a] and Section 1.4).

1.2.3 Contact Determination

The computation of forces between the virtual tool and other objects requires a model of collision response. Forces between the virtual objects must be computed from contact information. Determining whether two virtual objects collide (i.e., intersect) is not enough and additional information, such as penetration distance, contact points, contact normals, and so forth, need to be computed. Contact determination describes the process of obtaining the contact information necessary for collision response [Bar92].

For two interacting virtual objects, collision response can be computed as a function of object separation, with worst-case cost $O(mn)$, or penetration depth, with a complexity bound of $\Omega(m^3n^3)$, where m and n denote the geometric complexity of the two objects. But collision response can be applied at multiple *contacts* simultaneously. Given two objects A and B, contacts can be defined as pairs of intersecting triangles or pairs of triangles within a distance tolerance.

The number of pairs of intersecting triangles is $O(mn)$ in worst-case pathological cases and the number of pairs of triangles inside a tolerance can be $O(mn)$ even in practical cases. In Chapter 3, we discuss the existing techniques for determining the contact information in more detail.

The cost of contact determination depends mostly on factors such as the convexity of the interacting objects or the contact configuration. There is no direct connection between the polygonal complexity of the objects and the cost of contact determination but, as a reference, existing exact collision detection methods can barely execute contact queries for force feedback between pairs of objects with 1,000 triangles in complex contact scenarios [KOLM03] at force update rates of 1 kHz.

Contact determination becomes particularly expensive in the interaction between textured surfaces. Studies have been done on the highest texture resolution that can be perceived through cutaneous touch, but there are no clear results regarding the highest resolution that can be perceived kinesthetically through an intermediate object. It is known that, in the latter case, texture-induced roughness perception is encoded in vibratory motion [KL02]. Psychophysics researchers report that 1 mm textures are clearly perceivable and perceived roughness appears to be even greater with finer textures [LKHG00]. Based on Shannon's sampling theorem, a 10 cm $\times$ 10 cm plate with a sinusoidal texture of 1 mm in orthogonal directions is barely correctly sampled with 40,000 vertices. This measure gives an idea of the triangulation density required for capturing surface detail of complex textured objects. Note that the triangulation density may grow by orders of magnitude if the textures are not sinusoidal and/or if information about normals and curvatures is also needed.

1.3 PSYCHOPHYSICS OF HAPTICS

In the design of contact determination algorithms for haptic rendering it is crucial to understand the psychophysics of touch and to account for perceptual factors. The structure and behavior of human touch have been studied extensively in the field of psychology. The topics analyzed by researchers include characterization of sensory phenomena as well as cognitive and memory processes.

Haptic perception of physical properties includes a first step of stimulus registration and communication to the thalamus, followed by a second step of higher level processing. Perceptual measures can be originated by individual mechanoreceptors but also by the integration of inputs from populations of different sensory units [KL03]. Klatzky and Lederman [KL03] discuss object and surface properties that are perceived through the sense of touch (e.g., texture, hardness, and weight) and divide them between geometric and material properties. They also analyze active exploratory procedures (e.g., lateral motion, pressure, or unsupported holding) typically conducted by subjects in order to capture information about the different properties.

Knowing the exploratory procedure(s) associated with a particular object or surface property, researchers have studied the influence of various parameters on the accuracy and magnitude of sensory outputs. Perceptual studies on tactile feature detection and identification, as well as studies on texture or roughness perception are of particular interest for haptic rendering. In this section we summarize existing research on perception of surface features and perception of roughness, and then we discuss the issues associated with the interaction of visual and haptic modalities.

1.3.1 Perception of Surface Features

Klatzky and Lederman describe two different exploratory procedures followed by subjects in order to capture shape attributes and identify features and objects. In *haptic glance* [KL95], subjects extract information from a brief haptic exposure of the object surface. Then they perform higher level processing for determining the identity of the object or other attributes. In *contour following* [KL03], subjects create a spatiotemporal map of surface attributes, such as curvature, that serves as the pattern for feature identification. Contact determination algorithms attempt to describe the geometric interaction between virtual objects. The instantaneous nature of haptic glance [KL95] makes it strongly dependent on purely geometric attributes, unlike the temporal dependency of contour following.

Klatzky and Lederman [KL95] conducted experiments in which subjects were instructed to identify objects from brief cutaneous exposures (i.e., haptic glances). Subjects had an advanced hypothesis of the nature of the object. The purpose of the study was to discover how, and how well, subjects identify objects from brief contact. According to Klatzky and Lederman, during haptic glance a subject has access to three pieces of information: roughness, compliance, and local features. Roughness and compliance are material properties that can be extracted from lower level processing, while local features can lead to object identification by feature matching during higher level processing. In the experiments, highest identification accuracy was achieved with small objects, whose *extent* fit on a fingertip. Klatzky and Lederman concluded that large contact area helped in the identification of textures or patterns, although it was better to have a stimulus of a size comparable to or just slightly smaller than that of the contact area for the identification of geometric surface features. The experiments conducted by Klatzky and Lederman posit an interesting relation between feature size and contact area during cutaneous perception.

Okamura and Cutkosky [OC99, OC01] analyzed feature detection in robotic exploration, which can be regarded as a case of object–object interaction. They characterized geometric surface features based on the ratios of their curvatures to the radii of the robotic fingertips acquiring the surface data. They observed that a larger fingertip, which provides a larger contact area, can miss small geometric features.

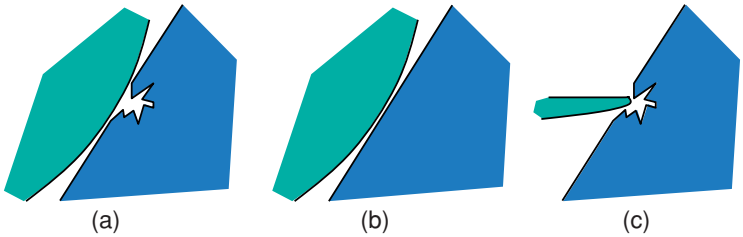

FIGURE 1.2: *Contact area and resolution.* (a) high-resolution model with large contact area; (b) low-resolution model with large contact area; (c) high-resolution model with small contact area [OL03b] (ACM, 2003).

To summarize, the studies by Klatzky and Lederman [KL95], and Okamura and Cutkosky [OC99, OC01] lead to the observation that human haptic perception of the existence of a geometric surface feature depends on the ratio between the contact area and the size of the feature, not the absolute size of the feature itself. The *size of a feature* is broadly defined here as width $\times$ length $\times$ height. For polygonal models, the width and length of a feature can be intuitively considered as the "inverse of resolution." That is, higher resolution around a local area implies that the width and length of the geometric surface features in that neighborhood are smaller, and vice versa. Given a polygonal model described at different levels of detail (LODs), the concept of "height" can be extended to describe the amount of surface deviation between LODs.

Fig. 1.2 illustrates the observation that relates contact area and perceptibility of features. The contact between two objects typically occurs along a certain contact area. With polygonal models, the contact area may be described by multiple contact points. The number of contact points grows if the objects are modeled at a higher resolution. Increasing the resolution beyond a sufficiently large value, however, may have little effect on the forces computed between the objects, because these forces are computed as a sum of contact forces arising from a net of contact points. One can argue that, intuitively, a larger contact area allows the objects to be described at a coarser resolution.

The conclusions drawn from perceptual studies set the basis for error metrics and multiresolution collision detection algorithms for haptic rendering [OL03a, OL03b] (described in detail in Section 3.6). The minimum acceptable resolution to represent an object will be governed by the relationship between surface deviation and contact area. Note that haptic error metrics differ notably from visual error metrics in the mesh simplification literature [Hop97, LE97] and from metrics of visual collision perception [OD01]. To contrast, the resolution required to represent an object for visual rendering is based on a combination of surface deviation (or Hausdorff distance) and the viewing distance to the object.

1.3.2 Perception of Texture and Roughness

Klatzky and Lederman [KL03] describe a textured surface as a surface with protuberant elements arising from a relatively homogeneous substrate. Interaction with a textured surface results in perception of roughness. Existing research on the psychophysics of texture perception indicates a clear dichotomy of exploratory procedures: (a) perception of texture with the bare skin and (b) perception through an intermediate (rigid) object, a probe.

Most of the research efforts have been directed toward the characterization of cutaneous perception of textures. Katz [Kat89] suggested that roughness is perceived through a combination of spatial and vibratory codes during direct interaction with the skin. More recent evidence demonstrates that static pressure distribution plays a dominant role in perception of coarse textures (features larger than 1 mm) [Led74, CJ92], but motion-induced vibration is necessary for perceiving fine textures [LS91, HR00]. As pointed out by Klatzky and Lederman [KL02], in object–object interaction roughness is encoded in vibratory motion transmitted to the subject.

In the last few years, Klatzky and Lederman have directed experiments that analyze the influence of several factors on roughness perception through a rigid probe. Klatzky et al. [KLH+03] distinguished three types of factors that may affect the perceived magnitude of roughness: inter-object physical interaction, skin- and limb-induced filtering prior to cutaneous and kinesthetic perception, and higher level factors such as efferent commands. The design of contact determination and collision response algorithms for haptic texture rendering is mostly concerned with factors related to the physical interaction between objects: object geometry [LKHG00, KLH+03], applied force [LKHG00], and exploratory speed [LKHR99, KLH+03]. The influence of these factors has been addressed in the design of haptic texture rendering algorithms [OJSL04], described in Chapter 4.

The experiments conducted by Klatzky and Lederman to characterize roughness perception [KL02] used a common setup: subjects explored a textured plate with a probe with a spherical tip and then they reported a subjective measure of roughness. Plates of jittered raised dots were used and the mean frequency of dot distribution was one of the variables in the experiments. The resulting data was analyzed by plotting subjective roughness values versus dot inter spacing in logarithmic graphs.

Klatzky and Lederman [KL99] compared graphs of roughness versus texture spacing (a) with finger exploration and (b) with a rigid probe. They concluded that, in the range of their data, roughness functions were best fit by linear approximations in finger exploration and by quadratic approximations in probe-based exploration. In other words, when perceived through a rigid spherical probe, roughness initially increases as texture spacing increases, but, after reaching a maximum roughness value, it decreases again. Based on this finding, the influence of other factors on roughness perception can be characterized by the maximum value of roughness and the value of texture spacing at which this maximum takes place.

Lederman et al. [LKHG00] demonstrated that the diameter of the spherical probe plays a crucial role in the maximum value of perceived roughness and the location of the maximum. The roughness peak is higher for smaller probes and it occurs at smaller texture spacing values. Lederman et al. [LKHG00] also studied the influence of the applied normal force during exploration. Roughness is higher for larger force, but the influence on the location of the peak is negligible. The effect of exploratory speed was studied by Lederman et al. [LKHR99]. They found that the peak of roughness occurs at larger texture spacing for higher speed. Also, with higher speed, textured plates feel smoother at small texture spacing and rougher at large spacing values. The studies reflected that speed has a stronger effect in passive interaction than in active interaction.

One conclusion that can be drawn from these studies is that the perception of roughness is intimately related to the trajectory traced by the probe. In particular, Klatzky and Lederman [KLH+03] identified the value of texture spacing at which the probe can exactly fall between two texture dots as *drop point* and they concluded that the peak of roughness perception occurs approximately at the drop point, and it depends on both geometric (i.e., probe diameter) and dynamic factors (i.e., speed).

1.3.3 Cross-Modal Interaction

It has been demonstrated several times that the combination of haptic feedback with other modalities may increase task performance in several situations, such as docking operations [BOYBK90], learning of pathways [Ins01], or early-age learning of scientific concepts [Dru97]. The sensory inputs to the haptic modality are processed along with sensory inputs of other modalities and the perceptual experiences are the result of all sensory inputs together. In the presentation of haptic rendering along with visual or auditory display, it is important to understand cross-modal interactions.

Klatzky and Lederman [KL03] discuss aspects of visual and haptic cross-modal integration from two perspectives: attention and dominance. Spence et al. [SPD00] have studied how visual and tactile cues can influence a subject's attention. Their conclusions are that visual and tactile cues are treated together in a single attentional mechanism and wrong attention cues can affect perception negatively.

Sensory dominance is usually studied by analyzing perceptual discrepancies in situations where cross-modal integration yields a unitary perceptual response. One example of relevance for this finding is the detection of object collision. During object manipulation, humans determine whether two objects are in contact based on a combination of visual and haptic cues. Early studies of sensory dominance seemed to point to a strong dominance of visual cues over haptic cues [RV64], but in the last decades psychologists agree that sensory inputs are weighted based on their statistical reliability or relative appropriateness, measured in terms of accuracy, precision,

and cue availability [HCGB99, EB01, KL03]. In some cases, apparent visual dominance can be beneficial for haptic rendering. For example, it is known that strong visual enforcement of non penetration constraints can enhance the haptic perception of stiffness [SBB96]. This property can be exploited to simulate hard non penetration constraints at a low update rate in the visual simulation, but looser constraints at a high update-rate in the haptic simulation.

The design of contact determination algorithms can also benefit from existing studies on the visual perception of collisions in computer animations. O'Sullivan and her colleagues [ORC99, OD01, ODGK03] have investigated different factors affecting visual collision perception, including eccentricity, separation, distractors, causality, and accuracy of simulation results. Basing their work on a model of human visual perception validated by psychophysical experiments, they demonstrated the feasibility of using these factors for scheduling interruptible collision detection among large numbers of visually homogeneous objects.

In some cases, haptic rendering is also presented along with auditory feedback, but the interaction between these two modalities is far less studied. Walker and Smith [WS85] found that subjects correlated high pitched sounds with small objects and, when this relationship was not matched, the interaction slowed down. Others have tried to exploit auditory feedback for reinforcing haptic display [MGC96], but have concluded that visual feedback is more effective, and the type of auditory stimuli must be selected very carefully in order to have a significant effect on haptic display.

1.4 STABILITY AND TRANSPARENCY

In haptic rendering, the human user is part of the dynamic system, along with the haptic device and the computer implementation of the virtual environment. The complete human-in-the-loop system can be regarded as a sampled-data system [CS94], with a continuous component (the user and the device) and a discrete one (the implementation of the virtual environment and the device controller). Stability becomes a crucial feature, because instabilities in the system can produce oscillations that distort the perception of the virtual environment, or uncontrolled motion of the device that can even hurt the user. In Section 1.2.2, we have briefly discussed the importance of stability for haptic rendering and we have introduced the effect of the force update rate on stability. In this section we review and discuss existing work in control theory related to stability analysis of haptic rendering in more detail.

1.4.1 Mechanical Impedance Control

The concept of mechanical impedance extends the notion of electrical impedance and refers to the quotient between force and velocity. Hogan [Hog85] introduced the idea of impedance control for contact tasks in manipulation. Earlier techniques controlled contact force, robot velocity, or both, but Hogan suggested controlling directly the mechanical impedance, which

governs the dynamic properties of the system. When the end effector of a robot touches a rigid surface, it suffers an instantaneous, drastic change of mechanical impedance, from low impedance in free space, to high impedance during contact. This phenomenon imposes serious difficulties on earlier control techniques, inducing instabilities.

The function of a haptic device is to display the feedback force of a virtual world to a human user. Haptic devices present control challenges very similar to those of manipulators for contact tasks. There are two major approaches of controlling a haptic device: impedance control and admittance control. In impedance control, the user moves the device, the rendering algorithm reads the position and orientation of the device, and the controller produces force and torque dependent on the interaction in the virtual world. In admittance control, the user applies force and torque to the device, which are read by the rendering algorithm, and the controller moves the device according to the virtual interaction.

In both impedance and admittance control, high control gains can induce instabilities. In impedance control, instabilities may arise in the simulation of stiff virtual surfaces. The device must react with large changes in force to small changes in the position. Conversely, in admittance control, rendering a stiff virtual surface is not a challenging problem, because it is implemented as a low controller gain. In admittance control, however, instabilities may arise during free-space motion in the virtual world, because the device must move at high velocities under small applied forces, or when the device rests on a stiff physical surface. Impedance and admittance control can therefore be regarded as complementary control techniques, best suited for opposite applications. Following the unifying framework presented by Adams and Hannaford [AH98a], contact determination and force computation algorithms are often independent of the control strategy.

1.4.2 Stable Rendering of Virtual Walls

Since the introduction of impedance control by Hogan [Hog85], the analysis of the stability of haptic devices and haptic rendering algorithms has focused on the problem of rendering stiff virtual walls. This was known to be a complex problem at early stages of research in haptic rendering [Kil76], but impedance control simplified the analysis, because a virtual wall can be modeled easily using stiffness and viscosity parameters.

Ouh-Young [OY90] created a discrete model of the Argonne ARM and the human arm and analyzed the influence of force update rate on the stability and responsiveness of the system. Minsky, Brooks, et al. [MOyS+90, BOYBK90] observed that update rates as high as 500 Hz or 1 kHz might be necessary in order to achieve stability.

Colgate and Brown [CB94] coined the term Z-width for describing the range of mechanical impedances that a haptic device can render while guaranteeing stability. They concluded that physical dissipation is essential for achieving stability and that the maximum achievable

virtual stiffness is proportional to the update rate. They also analyzed the influence of position sensors and quantization, and concluded that sensor resolution must be maximized and the velocity signal must be filtered.

Almost in parallel, Salcudean and Vlaar [SV94] studied haptic rendering of virtual walls, and techniques for improving the fidelity of the rendering. They compared a continuous model of a virtual wall with a discrete model that accounts for differentiation of the position signal. The continuous model is unconditionally stable, but this is not true for the discrete model. Moreover, in the discrete model fast damping of contact oscillations is possible only with rather low contact stiffness and, as indicated by Colgate and Brown [CB94] as well, this value of stiffness is proportional to the update rate. Salcudean and Vlaar proposed the addition of braking pulses, proportional to collision velocity, for improving the perception of virtual walls.

1.4.3 Passivity and Virtual Coupling

A subsystem is *passive* if it does not add energy to the global system. Passivity is a powerful tool for analyzing stability of coupled systems, because the coupled system obtained from two passive subsystems is always stable. Colgate and his colleagues were the first to apply passivity criteria to the analysis of stability in haptic rendering of virtual walls [CGSS93]. Passivity-based analysis has enabled separate study of the behavior of the human subsystem, the haptic device, and the virtual environment in force-feedback systems.

1.4.3.1 Human Sensing and Control Bandwidths

Hogan discovered that the human neuromuscular system exhibits externally simple, springlike behavior [Hog86]. This finding implies that the human arm holding a haptic device can be regarded as a passive subsystem, and the stability analysis can focus on the haptic device and the virtual environment.

Note that human limbs are not passive in all conditions, but the bandwidth at which a subject can perform active motions is very low compared to the frequencies at which stability problems may arise. Some authors [Shi92, Bur96] report that the bandwidth at which humans can perform controlled actions with the hand or fingers is between 5 and 10 Hz. On the other hand, sensing bandwidth can be as high as 20 to 30 Hz for proprioception, 400 Hz for tactile sensing, and 5 to 10 kHz for roughness perception.

1.4.3.2 Passivity of Virtual Walls

Colgate and Schenkel [CS94] observed that the oscillations perceived by a haptic user during system instability are a result of active behavior of the force-feedback system. This active behavior is a consequence of time delay and loss of information inherent in sampled-data systems, as suggested by others before [BOYBK90]. Colgate and Schenkel formulated passivity conditions in haptic rendering of a virtual wall. For that analysis, they modeled the virtual wall as a

viscoelastic unilateral constraint, and they accounted for the continuous dynamics of the haptic device, sampling of the position signal, discrete differentiation for obtaining velocity, and a zero-order hold of the output force. They reached a sufficient condition for passivity that relates the stiffness K and damping B of the virtual wall, the inherent damping b of the device, and the sampling period T:

$$b > \frac{KT}{2} + B. \tag{1.1}$$

1.4.3.3 Stability of Nonlinear Virtual Environments

After deriving stability conditions for rendering virtual walls modeled as unilateral linear constraints, Colgate and his colleagues considered more complex environments [CSB95]. A general virtual environment is nonlinear, and it presents multiple and variable constraints. Their approach enforces a discrete-time passive implementation of the virtual environment and sets a multidimensional viscoelastic *virtual coupling* between the virtual environment and the haptic display. In this way, the stability of the system is guaranteed as long as the virtual coupling is itself passive, and this condition can be analyzed using the same techniques as those used for virtual walls [CS94]. As a result of Colgate's virtual coupling [CSB95], the complexity of the problem was shifted towards designing a passive solution of virtual world dynamics. As noted by Colgate et al. [CSB95], one possible way to enforce passivity in rigid-body dynamics simulation is to use implicit integration with passive collision response models.

Adams and Hannaford [AH98a] provided a framework for analyzing stability with admittance-type and impedance-type haptic devices. They derived stability conditions for coupled systems based on network theory. They also extended the concept of virtual coupling to admittance-type devices. Miller et al. [MCF99] extended Colgate's passivity analysis techniques, relaxing the requirement of passive virtual environments but enforcing *cyclopassivity* of the complete system. Hannaford and his colleagues [HRK02] investigated the use of adaptive controllers, instead of the traditional fixed-value virtual couplings. They designed passivity observers and passivity controllers for dissipating the excessive energy generated by the virtual environment.

1.4.4 Multirate Approximation Techniques

Multirate approximation techniques, though simple, have been successful in improving the stability and transparency of haptic rendering systems. The idea is to perform a full update of the virtual environment at a low frequency (limited by computational resources and the complexity of the system) and to use a simplified approximation for performing high-frequency updates of force feedback.

Adachi [AKO95] proposed an *intermediate representation* for haptic display of complex polygonal objects. In a slow collision detection thread, he computed a plane that served as a unilateral constraint in the force-feedback thread. This technique was later adapted by Mark et al. [MRF+96], who interpolated the intermediate representation between updates. This approach enables higher stiffness values than approaches that compute the feedback force values at the rate determined by collision detection. More recently, a similar multirate approach has been followed by many authors for haptic interaction with deformable models [AH98b, cT00, DAK04a]. Ellis et al. [ESJ97] produce higher quality rendering by directly up-sampling the output force values. Multirate techniques for 6-DoF haptic rendering are covered in more detail in Section 2.4.

1.5 THREE-DOF HAPTIC RENDERING

Much of the existing work in haptic rendering has focused on 3-DoF haptic rendering [ZS95, RKK97, TJC97, GLGT99, HBS99] (see Fig. 1.3). Given a virtual object A and the 3D position of a point $\mathbf{p}$ governed by an input device, 3-DoF haptic rendering can be summarized as finding a contact point $\mathbf{p}'$ constrained to the surface of A. The contact force will be computed as a function of $\mathbf{p}$ and $\mathbf{p}'$. In a dynamic setting, assuming that A is a polyhedron with n triangles, the problem of finding $\mathbf{p}'$ has an $O(n)$ worst-case complexity. Using spatial partitioning strategies and exploiting motion coherence, however, the complexity becomes O(1) in many practical situations [GLGT99].

This reduced complexity has made 3-DoF haptic rendering an attractive solution for many applications with force feedback, such as: sculpting and deformation [DQ+99, GEL00, MQW01], painting [JTK+99, GEL00, BSLM01, FOL02, AWD+04], volume visualization [AS96], nanomanipulation [TRC+93], and training for diverse surgical

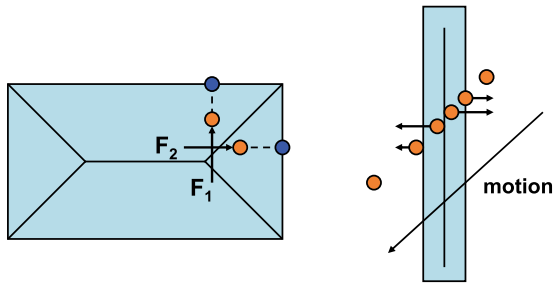

FIGURE 1.3: *Problems in 3-DoF haptic rendering*. Using closest features for penalty-based force computation produces force discontinuities. On the left, force discontinuity when switching between Voronoi regions. On the right, pop-through problem.

operations [KKH+97, GSM+97]. In each of these applications, the interaction between the subject and the virtual objects is sufficiently captured by a point–surface contact model.

This section briefly surveys some of the most popular 3-DoF haptic rendering techniques and special methods for rendering large data sets. The section concludes with a discussion on what applications are efficiently solved with 3-DoF haptic rendering and what applications require 6-DoF interaction. Three-DoF haptic texture rendering algorithms receive special treatment in Section 4.1.

1.5.1 Constraint-Based Methods

As mentioned earlier, 3-DoF haptic rendering methods compute feedback force as a function of the separation between the probe point controlled with the haptic device and a contact point constrained to the surface of the haptically rendered object. Early 3-DoF haptic rendering methods set the contact point as the point on the surface of the object closest to the probe point. As has been addressed by Zilles and Salisbury [ZS95], the closest point may jump arbitrarily along the surface. As a result and as shown in Fig. 1.3, closest-point methods lead to force discontinuities and possible "pop-through" problems, in which the contact point jumps between opposing sides of the object.

Zilles and Salisbury proposed the *god-object* method to solve the discontinuities induced by closest-point methods (see Fig. 1.4). Knowing the position of the haptic probe in the current frame and the previous frame, they identified the set of surfaces that constrained the inter-frame motion of the haptic probe. Given this set of active constraints, they computed the position of the contact point as a constrained optimization problem using Lagrange multipliers. More specifically, they defined a cost function based on the distance between the probe point and the contact point, and they added penalty terms due to the constraint surfaces, weighted by Lagrange multipliers. Then, they minimized the cost function and solved for the contact point. In practice, with Zilles and Salisbury's formulation the contact point may be constrained by one to three surfaces (i.e., plane constraint, concave edge constraint, or concave vertex constraint).

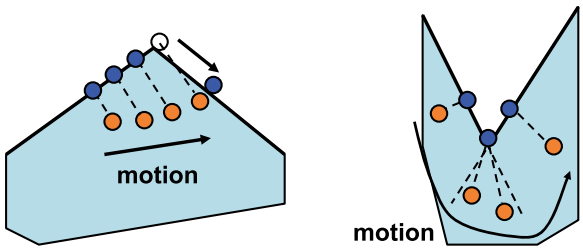

FIGURE 1.4: *Constraint-based method.* Behavior of the god-object method in convex (left) and concave (right) regions.

Finally, they computed feedback forces based on the distance between the probe point and the contact point.

Ruspini et al. [RKK97] followed a similar approach to Zilles and Salisbury, which they called *virtual proxy*. They modeled the contact point as a sphere of small radius and solved the optimization problem in the configuration space. As opposed to the *god-object* method, they also formulated the problem of identifying the set of active constraints from the local neighborhood in configuration space. At each frame, Ruspini et al. set as goal the position of the probe point. They identified possible constraint surfaces using the ray between the old position of the *virtual proxy* and the goal position. Then, they solved a quadratic optimization problem and found a subgoal position. They repeated this process until the subgoal position was fully constrained. Ruspini et al. realized that the quadratic optimization problem for a spherical proxy was a convex one and could be solved efficiently in the dual space defined by the normals of the constraint planes. Ruspini and his colleagues also added other effects, such as force shading for rounding of corners (by modifying the normals of constraint planes) or friction (by adding dynamic behavior to the contact point).

1.5.2 Scalable Collision Detection for Three-DoF Haptic Display

When haptically rendering large environments or data sets, the rendering algorithms face two major problems: memory limitations and an exorbitant computational cost of collision-detection routines. In 3-DoF haptic rendering, however, the high spatial coherence of the collision queries can be exploited to design effective rendering algorithms. A number of researchers have followed this research direction.

Gregory et al. [GLGT99] presented *H-Collide*, a fast collision detection algorithm for 3-DoF haptic rendering (see Fig.1.5). As described in the previous section, constraint-based 3-DoF haptic rendering methods rely on ray–triangle intersections for identifying active constraint planes. *H-Collide* constructs a hybrid hierarchical representation of the scene's geometry as a pre-processing step. First, it partitions the scene geometry based on a uniform grid. Then, for each cell in the grid, it computes an oriented-bounding-box hierarchy (OBB-Tree) of the triangles that lie in that cell. Lastly, it stores pointers to the OBB-Trees using a hashing technique. The ray–triangle intersection tests are performed in two steps. In the first step, the ray is tested for collision with the grid cells using a hash table. This returns a set of OBB-Trees. In the second step, the ray is tested for intersection with the OBBs and, if necessary, against triangles stored in the leaves of the OBB-Trees. The high spatial and temporal coherence of 3-DoF haptic rendering makes spatial partitioning an excellent and convenient culling approach, as the ray often does not need to be tested for intersection against surface areas that are far from the region where the haptic probe is exploring.

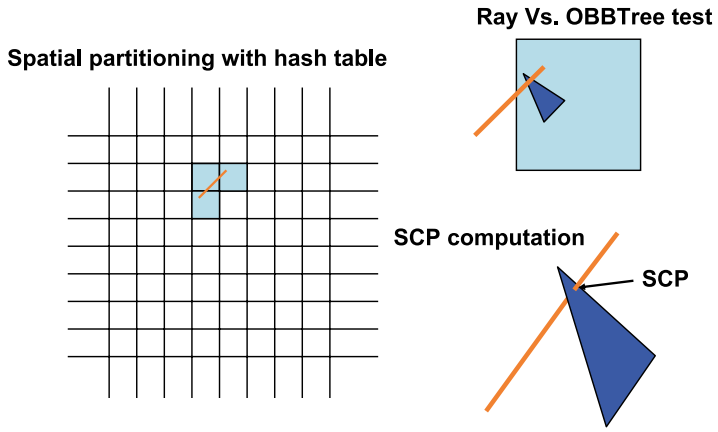

FIGURE 1.5: *H-Collide method.* Schematic pipeline of the H-Collide collision-detection method. Potential contacts are first localized using a spatial partitioning approach. Then ray-OBB intersections are performed, and the surface collision point (SCP) is located.

Others have addressed the problem of haptically rendering large terrain data sets. The methods proposed by Walker and Salisbury [WS03] and Potter et al. [PJC04] both rely on the assumption that the geometric data to be rendered is a height field. The proxy graph algorithm presented by Walker and Salisbury [WS03] restricts the position of the proxy contact point to remain on the edges and vertices of the rendered geometric surface. With this limitation, they achieve significant speedups in the collision detection process and they are able to sweep over a large number of surface triangles on one haptic frame. The rendering algorithm presented by Potter et al. [PJC04] describes the height field data as a bilinear patch instead of a triangulated surface. Ray surface intersection tests and closest-point search operations are optimized in the situation where the surface can be simply described by its height value.

Similarly, Glencross et al. [GHL05] have proposed a caching technique for haptic rendering of up to hundreds of thousands of distinct objects in the same scene. Their haptic cache maintains only objects in the locality of the haptic probe.

1.5.3 Three-DoF Versus Six-DoF Haptic Rendering

As mentioned earlier, 3-DoF haptic display is adequate for applications where the interaction between the subject and the virtual objects is sufficiently captured by a point–surface contact model. However, for 6-DoF manipulation and exploration where a subject grasps an object and touches other objects in the environment, the interaction generally cannot be modeled by a point–surface contact. One reason is the existence of multiple contacts that impose multiple simultaneous non penetration constraints on the virtual tool. In a simple 6-DoF manipulation

example, such as the insertion of a peg in a hole, the virtual tool (i.e., the peg) collides at multiple points with the rest of the scene (i.e., the walls of the hole and the surrounding surface). This contact configuration cannot be modeled as a point–object contact. Another reason is that the virtual tool presents six DoFs, three for translation and three for rotation, as opposed to the three DoFs of a point. The feasible trajectories of the peg are embedded in a six-dimensional space with translational and rotational constraints that cannot be captured with three DoFs.

Note that some cases of object–object interaction have been modeled in practice by ray–surface contact [BHS97]. In particular, several surgical procedures are performed with 4-DoF tools (e.g., laparoscopy), and this reduced number of DoFs has been exploited in training simulators with haptic feedback [cTS02]. Nevertheless, these approximations are valid only in a limited number of situations and cannot capture full 6-DoF object manipulation.

1.6 CHAPTER OUTLINE

In Chapter 2, we present general methodologies for 6-DoF haptic rendering. As already discussed, haptic rendering requires higher update rates than visual rendering, thereby limiting the complexity of the models that can be displayed. The increasing geometric complexity of digital models further widens the gap between the models that can be graphically displayed at interactive rates and those that can be haptically displayed. Two general classes of haptic rendering methodologies have been proposed to address this problem. We will describe them in detail in Chapter 2.

Collision detection is one of the foremost computational bottlenecks for achieving effective haptic rendering of complex environments. Due to its importance, we have devoted a significant portion of this synthesis to discuss the topic in Chapter 3. Static and dynamic factors such as large contact areas or high impact velocities often increase the cost of collision detection, and in this chapter we will cover data structures and algorithms that try to minimize the cost of collision detection for haptic rendering.

In designing contact determination algorithms for high-fidelity haptic rendering, it is crucial to understand the psychophysics of touch and to account for perceptual factors. Psychophysics studies indicate that some of the factors that increase the cost of collision detection may at the same time limit our haptic ability to perceive the influence of high-resolution geometric details. These perceptual limitations thereby motivate the use of approximate collision detection algorithms that adaptively select the appropriate geometric resolution for contact computations. In Section 3.6, we show how classic approximate techniques from computer graphics, such as levels of detail, can be adapted to the problem of collision detection. *Contact levels of detail* constitute a data structure that integrates in one dual hierarchy level-of-detail surface representations and bounding volume hierarchies for accelerating collision detection. A sensation-preserving simplification of geometric detail, along with error metrics for haptic

rendering, enable the selection of the appropriate level of detail at each potential contact location independently.

Next, in Chapter 4, we present several techniques for haptic texture rendering. *Haptic textures* decouple the description of objects with high-resolution features into coarse parameterized meshes and texture images that store the fine geometric detail. Using haptic textures, collision detection can be formulated as a two-step problem that first identifies contact locations using coarse meshes and then refines contact information accounting for the effect of geometric details. Along with perceptually driven force models, haptic textures can enable 6-DoF haptic rendering of the interaction between two complex textured models.

As we examine the recent developments in haptic rendering and its applications, we feel that computational haptics, a human interface technology still in its early stages, will significantly improve and enrich human–computer interaction by engaging one of our most basic sensory channels—the sense of touch. This synthesis concludes by examining some open research challenges and opportunities in the field.

CHAPTER 2

Six-DoF Haptic Rendering Methodologies

As introduced in Section 1.2.1, the existing methods for haptic rendering can be classified into two categories based on their overall approaches: *direct rendering* methods and *simulation-based* methods. We start this chapter with a brief description of rigid-body simulation techniques, focusing the discussion on the *collision response* methods that have been applied to 6-DoF haptic rendering. We then present in detail the two main rendering approaches and list specific techniques in each category and describe some of their uniqueness from others. We end the chapter with a discussion on multirate rendering techniques.

2.1 RIGID-BODY SIMULATION

Computation of the motion of a rigid body consists of solving a set of ordinary differential equations (ODEs). The most common way to describe the motion of a rigid body is by means of the Newton–Euler equations, which define the time derivatives of the linear momentum, **P**, and angular momentum, **L**, as a function of external force, **F**, and torque, **T**:

$$\begin{aligned} \mathbf{F}(t) &= \dot{\mathbf{P}}(t) = m\,\ddot{\mathbf{x}}(t), \\ \mathbf{T}(t) &= \dot{\mathbf{L}}(t) = \omega(t) \times (M\omega(t)) + M\dot{\omega}(t). \end{aligned} \tag{2.1}$$

As shown in the equations, momentum derivatives can be expressed in terms of the linear acceleration of the center of mass $\ddot{\mathbf{x}}$, the angular velocity ω, the mass of the body m, and the mass matrix M.

The complexity of rigid-body simulation lies in the computation of force and torque resulting from contacts between bodies. Research in the field of rigid-body simulation has mostly centered around different methods for computing contact forces and the resulting accelerations and velocities, ranging from approximate methods that consider each contact independently (such as penalty-based methods) to analytic methods that concurrently account for all nonpenetration constraints. Important efforts have been devoted to capturing friction forces as well.

This section briefly describes the main methods for solving the motion of colliding rigid bodies, focusing on their applicability to haptic rendering. For further information, please refer to Baraff's and Mirtich's dissertations [Bar92, Mir96], SIGGRAPH course notes on the topic [BW01], or recent work by Stewart and Trinkle [ST00]. In the last few years, especially in the field of computer graphics, attention has been drawn toward the problem of simulating the interaction of many rigid bodies [Mir00, MS01, GBF03, KEP05]. However, in many practical applications of 6-DoF haptic rendering (e.g., assembly and disassembly tasks or surgical operations on bones or hard structures), the environment can be considered as static and one is mostly concerned with the dynamics of the virtual tool. Therefore, the interaction of many rigid objects is not discussed here.

2.1.1 Implicit Integration of Rigid-Body Dynamics

As described in Section 1.4.3, Colgate et al. [CSB95] pointed out that implicit integration of the differential equations describing the virtual environment can ease the design of a stable haptic display. Otaduy and Lin [OL05] followed this observation to simulate the motion of the virtual tool by discretizing the Newton–Euler equations of rigid-body motion using a semi-implicit Euler method. Moreover, implicit integration enhances display transparency by enabling stable simulation of the virtual tool with small mass values. Implicit time-stepping schemes have also been used for rigid-body simulation in other applications, both with penalty-based methods [Wu00] and with constraint-based methods [ST00].

The state $\mathbf{y}$ of a rigid body can be formulated in terms of 13 variables: three for the position of its center of mass, $\mathbf{x}$, four for a quaternion describing its orientation, $\mathbf{q}$, three for its linear momentum, $\mathbf{P}$, and three for its angular momentum, $\mathbf{L}$. With this selection of state variables, the Newton–Euler equations (2.1) yield the following ODEs:

$$\dot{\mathbf{y}}(t) = \begin{pmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{q}} \\ \dot{\mathbf{P}} \\ \dot{\mathbf{L}} \end{pmatrix} = \begin{pmatrix} \frac{1}{m}\mathbf{P} \\ \frac{1}{2}\boldsymbol{\omega}_{\mathrm{q}}\mathbf{q} \\ \mathbf{F} \\ \mathbf{T} \end{pmatrix} = \mathbf{f}(\mathbf{y}, t), \tag{2.2}$$

where m is the mass of the body. The term $\boldsymbol{\omega}_{\mathrm{q}}$ indicates a quaternion with scalar part 0 and vector part the angular velocity $\boldsymbol{\omega}$. Given the mass matrix M of the body, computed in a local frame, and the rotation matrix R from the world frame to the local frame of the body, the angular velocity $\boldsymbol{\omega}$ can be expressed in terms of state variables as

$$\boldsymbol{\omega} = RM^{-1}R^{T}\mathbf{L}. \tag{2.3}$$

Assuming the virtual tool is the only moving body, the state vector $\mathbf{y}$ reduces to the 13 state variables of one rigid body. The external forces (and similarly for the torques) may comprise

the weight of the object, contact forces, and possibly a user coupling force (e.g., with virtual coupling).

2.1.1.1 Semi-Implicit Euler Discretization

Implicit discretization of the ODEs using the backward Euler formula yields the following state update:

$$\mathbf{y}_n = \mathbf{y}_{n-1} + \Delta t \dot{\mathbf{y}}_n. \tag{2.4}$$

Substituting Eq. (2.2) in Eq. (2.4) leads to a nonlinear equation in the state variables **x**, **q**, **P**, and **L**. A nonlinear solver, such as Newton's method, can be used for finding the exact solution to this system. However, one can trade numerical accuracy for desired speed and linearly approximate Eq. (2.4) using the Taylor expansion of **f**. This approximation leads to a semi-implicit backward Euler discretization, in which $\frac{\partial \mathbf{f}}{\partial \mathbf{y}}$ is the Jacobian of the equations of rigid-body motion. Rearranging terms, the linear system of equations can be expressed in the form

$$\left(I - \Delta t \frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right)(\mathbf{y}_n - \mathbf{y}_{n-1}) = \Delta t \mathbf{f}_{n-1}. \tag{2.5}$$

Under the assumption that the virtual tool is the only moving object, $\left(I - \Delta t \frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right)$ is a 13×13 dense and nonsymmetric matrix. The linear system can be solved by Gaussian elimination. The remaining of this section focuses on the formulation of the Jacobian $\frac{\partial \mathbf{f}}{\partial \mathbf{y}}$.

2.1.1.2 Jacobian of the Equations of Motion

The Jacobian of Eq. (2.2) can be expressed as

$$\frac{\partial \mathbf{f}}{\partial \mathbf{y}} = \begin{pmatrix} 0 & 0 & \frac{1}{m}I & 0 \\ 0 & \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{q}} & 0 & \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{L}} \\ \frac{\partial \mathbf{F}}{\partial \mathbf{x}} & \frac{\partial \mathbf{F}}{\partial \mathbf{q}} & \frac{\partial \mathbf{F}}{\partial \mathbf{P}} & \frac{\partial \mathbf{F}}{\partial \mathbf{L}} \\ \frac{\partial \mathbf{T}}{\partial \mathbf{x}} & \frac{\partial \mathbf{T}}{\partial \mathbf{q}} & \frac{\partial \mathbf{T}}{\partial \mathbf{P}} & \frac{\partial \mathbf{T}}{\partial \mathbf{L}} \end{pmatrix}. \tag{2.6}$$

The expression of the derivative of orientation, $\dot{\mathbf{q}}$, is highly nonlinear and leads to two nonzero blocks in the Jacobian, as shown in Eq. (2.6). Given a quaternion $\mathbf{q} = (x, y, z, s)$, the expression of $\dot{\mathbf{q}}$ can be rewritten as a matrix–vector multiplication:

$$\dot{\mathbf{q}} = 1/2\omega_q \mathbf{q} = Q\omega, \qquad Q = \frac{1}{2}\begin{pmatrix} s & z & -y \\ -z & s & x \\ y & -x & s \\ -x & -y & -z \end{pmatrix}. \tag{2.7}$$

The combination of Eqs. (2.3) and (2.7) yields the following Jacobians:

$$\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{L}} = QRM^{-1}R^T, \tag{2.8}$$

$$\frac{\partial \dot{\mathbf{q}}}{\partial q_i} = \frac{\partial Q}{\partial q_i}\boldsymbol{\omega} + Q\frac{\partial \boldsymbol{\omega}}{\partial q_i}, \tag{2.9}$$

$$\frac{\partial \boldsymbol{\omega}}{\partial q_i} = \left(\frac{\partial R}{\partial q_i}M^{-1}R^T + RM^{-1}\frac{\partial R}{\partial q_i}^T\right)\mathbf{L}. \tag{2.10}$$

Note that $\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{q}}$ is expressed separately for each of the components q_i of $\mathbf{q}$.

The evaluation of $\frac{\partial \mathbf{f}}{\partial \mathbf{y}}$ also requires the Jacobians of the equations of external forces (and torques). More details on implicit integration of rigid-body dynamics for haptic rendering can be found in [OL05].

2.1.2 Penalty-Based Methods

One method for implementing collision response is the insertion of stiff springs at the points of contact [MW88]. This method is inspired by the fact that, when objects collide, small deformations take place at the region of contact, and these deformations can be modeled with springs, even if the objects are geometrically rigid.

Given two intersecting objects A and B, penalty-based collision response requires the definition of a contact point $\mathbf{p}$, a contact normal $\mathbf{n}$, and a penetration depth δ. The penalty-based spring force and torque applied to object A are defined as follows:

$$\begin{aligned}\mathbf{F}_A &= -f(\delta)\mathbf{n},\\ \mathbf{T}_A &= (\mathbf{p}-\mathbf{x})\times\mathbf{F}_A,\end{aligned} \tag{2.11}$$

where $\mathbf{x}$ is the position of the center of mass of A. Opposite force and torque are applied to object B. The function f could be a linear function defined by a constant stiffness k or a more complicated nonlinear function. It could also contain a viscous term, dependent on the derivative of the penetration depth.

The basic formulation of penalty methods can be modified slightly in order to introduce repulsive forces between objects, by inserting contact springs when the objects come closer than a distance tolerance d. In this way, object interpenetration occurs less frequently. The addition of a tolerance has two major advantages: the possibility of using penalty-based methods in applications that do not allow object interpenetration and a reduction of the cost of collision detection. As noted in Section 3.4, computation of penetration depth is notably more costly than computation of separation distance.

The general penalty-based response of Eq. (2.11) can be rewritten more specifically as a viscoelastic penalty-based response accounting for tolerance d. For simplicity, we assume a

point-on-plane contact between the virtual tool A and a static environment object B, with contact normal $\mathbf{n}$ and contact points $\mathbf{p} \in A$ and $\mathbf{p}_0 \in B$. Note that the contact point $\mathbf{p}$ can also be expressed in local coordinates of the virtual tool as $\mathbf{p} = \mathbf{x} + R\mathbf{r}$. Then, the penalty-based contact-force model is

$$\begin{aligned} \mathbf{F}_A &= -kN(\mathbf{x} + R\mathbf{r} - \mathbf{p}_0) - k\, d\, \mathbf{n} - bN(\mathbf{v} + \boldsymbol{\omega} \times (R\mathbf{r})), \\ \mathbf{T}_A &= (R\mathbf{r}) \times \mathbf{F}_A. \end{aligned} \tag{2.12}$$

N is a matrix that projects a vector onto the normal of the constraint plane, and it is computed as $\mathbf{n}\,\mathbf{n}^T$.

Penalty-based methods offer several attractive properties: the force model is local to each contact and computationally simple, object interpenetration is inherently allowed, and the cost of the numerical integration is almost insensitive to the complexity of the contact configuration. This last property makes penalty-based methods best suited for interactive applications with fixed time steps. In fact, penalty-based methods have been applied to game physics [Wu00, Lar01] and many 6-DoF haptic rendering approaches [MPT99, KOLM03, JW03, OL05].

However, penalty-based methods also have some disadvantages. There is no direct control over physical parameters, such as the coefficient of restitution. Geometric discontinuities in the location of contact points may lead to torque discontinuities, as depicted schematically in Fig. 2.1. Nonpenetration constraints are enforced by means of very high contact stiffness and this circumstance leads to instability problems if numerical integration is executed using fast, explicit methods.

The solution of penalty-based simulation using implicit integration, however, enhances stability in the presence of high contact stiffness ([Wu00, Lar01]). Otaduy and Lin [OL05] formulated the Jacobians of viscoelastic penalty-based response for contact between the virtual tool and a static environment, following Eq. (2.12).

Friction effects can be incorporated into penalty-based methods by means of localized force models that consider each contact point independently. Most local friction methods propose different force models for static or dynamic situations [Kar85, HA00]. Static friction is

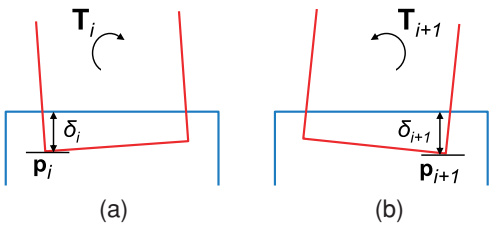

FIGURE 2.1: *Torque discontinuity.* (a) Penetration depth and torque at time t_i, with the contact point $\mathbf{p}_i$; (b) penetration depth and torque at time t_{i+1}, after the contact moves to the contact point $\mathbf{p}_{i+1}$.

modeled by fixing adhesion points on the surfaces of the colliding objects and setting tangential springs between the contact points and the adhesion points. If the elastic friction force becomes larger than a threshold determined by the normal force and the friction coefficient, the system switches to dynamic mode. In the dynamic mode, the adhesion point follows the contact point. The system returns to static mode if the velocity falls below a certain threshold.

2.1.3 Contact Clustering

In the previous section, we have analyzed penalty-based collision response by considering individual contacts. However, the output of the contact determination step (see Chapter 3) has a strong influence on the smoothness of collision response and, as a result, on the stability of numerical integration. As pointed out by Larsen [Lar01], when a new contact point is added, the associated spring must be unstretched. In other words, the penetration depth value must be zero initially and must grow smoothly. The existence of geometry-driven discontinuities is an inherent problem of penalty-based simulations with fixed time steps. These discontinuities induce continuous-time models that are no longer passive and implicit integration is not sufficient for achieving discrete-time passivity. One possible solution would be to sequentially activate continuous-time passive local force models [MH05], although this approach has so far been applied only to deformable models.

Luo and Xiao [LX04] propose geometric and dynamic rules for determining a minimum set of active contacts, their configuration, and collision forces, but their approach has not yet been tested on complex polygonal models.

2.1.3.1 Contact Averaging

McNeely et al. [MPT99] suggested limiting the total stiffness after reaching a certain number of contacts (typically 10). Given n contacts and a collision force $\mathbf{F}_{\text{total}}$, the net force $\mathbf{F}_{\text{net}}$ applied to the virtual tool is

$$\begin{cases} \mathbf{F}_{\text{net}} = \mathbf{F}_{\text{total}}, \text{ if } & n < 10 \\ \mathbf{F}_{\text{net}} = \frac{\mathbf{F}_{\text{total}}}{n/10}, \text{ if } & n \geq 10 \end{cases}. \tag{2.13}$$

2.1.3.2 Proximity-Based Clustering

Kim et al. [KOLM03] proposed a proximity-based clustering technique to reduce the number of representative contacts for collision response. Contacts that are closer than a threshold ε from each other are grouped and averaged to a single contact. Hence, given a contact cluster S and a contact C_j with contact point $C_j.\mathbf{p}$,

$$C_j \in S \iff \exists C_k \in S \quad \text{such that} \quad \|C_k.\mathbf{p} - C_j.\mathbf{p}\| < \epsilon. \tag{2.14}$$

In practice, they accelerated the clustering operations using an octree data structure and slightly relaxing the proximity threshold. First, contacts are inserted in the appropriate cell in the octree, whose leaves are of size ϵ. The clusters are initialized by grouping all the contacts in the same leaf cell. Next, adjacent nonempty clusters are merged together. Following this procedure, the clustering policy is uncertain if the distance between the contacts is in the range $(\varepsilon, 2\sqrt{3}\varepsilon)$. At termination of the clustering procedure, the contact data (i.e., position, normal, and/or penetration depth) of all contacts in a cluster are averaged to produce a representative contact.

2.1.3.3 *K*-Means Clustering

Otaduy and Lin [OL05] improved on the contact clustering approach of Kim et al. [KOLM03] by preventing problems associated with the variability of the number of contacts. Their clustering methods is based on the K-means clustering technique [JMF99], limits the total translational stiffness applied to the virtual tool, and provides spatial filtering of contact data for densely sampled objects, which is useful for alleviating discontinuities.

Given a set of n contacts $\{C_0, C_1, \ldots, C_{n-1}\}$ and K clusters $\{S_0, S_1, \ldots, S_{K-1}\}$, each cluster contains a representative contact. If the number of input contacts is $n < K$, only n clusters are created. The clusters are defined implicitly by storing an additional parameter along with each contact C: the cluster it belongs to, S. Then, a contact C is defined as a tuple $(\mathbf{p}, \mathbf{p}_0, \mathbf{n}, \delta, S)$. In the description of the clustering algorithm, each parameter of a contact is referenced as $C.parameter$ (e.g., $C.\mathbf{p}$). Similarly, a cluster S is defined as a tuple $(\mathbf{p}, \mathbf{p}_0, \mathbf{n}, \delta)$, where $\mathbf{p}$, $\mathbf{p}_0$, $\mathbf{n}$, and δ are the contact parameters of the cluster representative.

The cost function f for the K-means clustering problem is formulated based on the Euclidean distance between each contact point $C.\mathbf{p}$ and the representative of the cluster it belongs to, $C.S.\mathbf{p}$, weighted by the penetration depth of the contact, $C.\delta$. This strategy increases the smoothness of penalty-based collision response. Specifically, the cost function f is written as

$$f = \frac{\sum_i^{n-1} \left((C_i.\delta + d) \| C_i.\mathbf{p} - C_i.S.\mathbf{p} \|^2 \right)}{\sum_i^{n-1} (C_i.\delta + d)}. \tag{2.15}$$

This cost function is minimized when the cluster representatives are located at the centroids of the clusters. This property is exploited by Lloyd's method [Llo57], a greedy algorithm that solves the K-means clustering problem by interleaving one step of centroid computation with one step of reclustering until the clusters converge. Lloyd's method can be adopted for computing contact clusters, as the clustering is expected to converge rapidly by exploiting temporal coherence and initializing cluster centroids at the positions of representative contacts from the previous frame. At every iteration of Lloyd's method, each contact is reassigned to its closest representative and the position of the representative of each cluster is recomputed as

the centroid of all the contact points in the cluster, weighted by their penetration depth. The expression for the position of each representative is

$$S.\mathbf{p} = \frac{\sum_{i,\ C_i.S=S} \left((C_i.\delta + d)C_i.\mathbf{p}\right)}{\sum_{i,\ C_i.S=S} (C_i.\delta + d)}. \tag{2.16}$$

Once the clusters converge, the remaining parameters of the representative contact for each cluster (i.e., $\mathbf{p}_0$, δ, and $\mathbf{n}$) can be computed based on the following expressions:

$$S.\delta = \frac{\sum_{i,\ C_i.S=S} ((C_i.\delta + d)C_i.\delta)}{\sum_{i,\ C_i.S=S} (C_i.\delta + d)}, \tag{2.17}$$

$$S.\mathbf{n} = \frac{\hat{\mathbf{n}}}{\|\hat{\mathbf{n}}\|},$$

$$\hat{\mathbf{n}} = \frac{\sum_{i,\ C_i.S=S} ((C_i.\delta + d)C_i.\mathbf{n})}{\sum_{i,\ C_i.S=S} (C_i.\delta + d)}, \tag{2.18}$$

$$S.\mathbf{p}_0 = S.\mathbf{p} - S.\delta(S.\mathbf{n}). \tag{2.19}$$

2.1.4 Impulse-Based Dynamics

Mirtich [MC95, Mir96] presented a method for handling collisions in rigid-body dynamics simulation based solely on the application of impulses to the objects. In situations of resting, sliding, or rolling contact, constraint forces are replaced by trains of impulses. Mirtich defined a collision matrix that relates contact impulse to the change in relative velocity at the contact. His algorithm decomposes the collision event into two separate processes: compression and restitution. Each process is parameterized separately, and numerical integration is performed in order to compute the velocities after the collision. The parameterization of the collision event enables the addition of a friction model to instantaneous collisions.

In the time-stepping engine of impulse-based dynamics, numerical integration must be interrupted before interpenetration occurs and valid velocities must be computed. One of the problems of impulse-based dynamics emerges during inelastic collisions from the fact that accelerations are not recomputed. The energy loss induced by a train of inelastic collisions reduces the time between collisions and increases the cost of simulation per frame. In order to handle this problem, Mirtich suggested the addition of unrealistic, but visually imperceptible, energy to the system when the microcollisions become too frequent. As has been pointed out by Mirtich, impulse-based approaches are best suited for simulations that are collision intensive, with multiple, different impacts occurring frequently.

Chang and Colgate [CC97] suggested the integration of passive impulse-based collision response in haptic rendering. The main problem in the context of haptic rendering is the need

to enforce a fixed-time stepping scheme. Chang and Colgate analyzed several possibilities for defining the contact state before applying collision response, with the conditions of collision resolution and passivity. Due to the lack of a solution for resting contacts, they proposed the possible combination with force-based methods (e.g., penalty forces). More recently, Constantinescu et al. [CSC04, CSC05] have also proposed the combination of penalty forces with impulsive response. A state machine is required in order to determine the state of the object under collision and has been successfully demonstrated in polygonal environments of low complexity. Constantinescu et al. have also proven the passivity of multiple impulses applied simultaneously using Newton's restitution law.

2.1.5 Constraint-Based Simulation

Constraint-based methods for the simulation of rigid-body dynamics handle all concurrent contacts in a single computational problem and attempt to find contact forces that produce physically and geometrically valid motions. Specifically, they integrate the Newton–Euler equations of motion (see Eq. (2.1)), subject to geometric constraints that prevent object interpenetration.

Here we outline the two main formulations in the literature: acceleration-based and velocity-based. In Section 2.3 we describe some example applications of constraint-based rigid-body simulation to 6-DoF haptic rendering.

2.1.5.1 Acceleration-Based Formulation

In acceleration-based formulations, the numerical integration of the Newton–Euler equations must be interrupted before objects interpenetrate. At a collision event, object velocities and accelerations must be altered, so that nonpenetration constraints are not violated and numerical integration can be restarted. One must first compute contact impulses that produce constraint-valid velocities. Then, one must compute contact forces that produce valid accelerations.

The relative normal accelerations $\mathbf{a}$ at the points of contact can be expressed as linear combinations of the contact forces $\mathbf{F}$. Moreover, one can impose nonpenetration constraints on the accelerations and nonattraction constraints on the forces:

$$\begin{aligned} &\mathbf{a} = \mathcal{A}\mathbf{F} + \mathbf{b}, \\ &\mathbf{a} \geq 0, \qquad \mathbf{F} \geq 0. \end{aligned} \tag{2.20}$$

Baraff [Bar89] pioneered the application of constraint-based approaches to rigid-body simulation in computer graphics. He posed constrained rigid-body dynamics simulation as a quadratic programming problem on the contact forces and proposed a fast, heuristic-based solution for the frictionless case. He defined a quadratic cost function based on the fact that contact forces occur only at contact points that are not moving apart:

$$\min\left(\mathbf{F}^T\mathbf{a}\right) = \min\left(\mathbf{F}^T\mathcal{A}\mathbf{F} + \mathbf{F}^T\mathbf{b}\right). \tag{2.21}$$

The quadratic cost function suggested by Baraff indicates that either the normal acceleration or the contact force should be zero at a resting contact. As indicated by Cottle et al. [CpS92], this condition can be formulated as a linear complementarity problem (LCP). Baraff [Bar91, Bar92] added dynamic friction to the formulation of the problem and suggested approaches for static friction, as well as a solution following an algorithm by Lemke [Lem65] with expected polynomial cost in the number of constraints. Earlier, Lötstedt had studied the problem of rigid-body dynamics with friction in the formulation of the LCP [L̈84]. Later, Baraff [Bar94] adapted an algorithm by Cottle and Dantzig [CD68] for solving frictionless LCPs to the friction case and achieved linear-time performance in practice.

2.1.5.2 Velocity-Based Formulation

Stewart and Trinkle [ST96, ST00] presented an implicit LCP formulation of constraint-based problems. Unlike previous algorithms, which enforced the constraints only at the beginning of each time step, their algorithm solves for contact impulses that also enforce the constraints at the end of the time step. The key difference is to formulate the geometric constraints directly on the velocities and establish the relationship between velocities and forces (or impulses) by incorporating a discrete approximation of accelerations. Their formulation eliminates the need to locate collision events, but it increases the number of constraints to be handled. A combination of shock propagation [GBF03] and constraint-correction techniques [Erl04] can be adopted for dealing with multiple contacts.

Stewart and Trinkle [ST96] mention the existence of geometry-driven discontinuities, similar to the ones appearing with penalty-based methods, in their implicit formulation of the LCP. After numerical integration of object position and velocities, new nonpenetration constraints are computed. If numerical integration is not interrupted at collision events, the newly computed nonpenetration constraints may not hold. Constraint violation may produce unrealistically high contact impulses and object velocities in the next time step. Stewart and Trinkle suggest solving a nonlinear complementarity problem, with additional cost involved.

2.2 DIRECT RENDERING

The overall architecture of direct rendering methods is shown in Fig. 2.2. Direct rendering relies on an impedance-type control strategy. First, the position and orientation of the haptic device are received from the controller and they are assigned directly to the virtual tool. Collision detection is then performed between the virtual tool and the environment. Collision response is typically computed as a function of object separation or penetration depth using penalty-based methods. Finally, the resulting contact force and torque are directly fed back to the device controller.

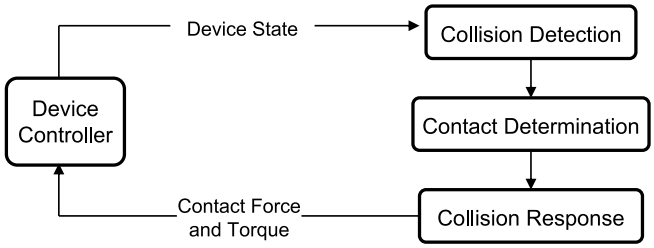

FIGURE 2.2: Direct rendering architecture.

The main advantage of direct rendering methods is that they are purely geometric and there is no need to simulate the rigid-body dynamics of the virtual tool. However, penetration values may be quite large and visually perceptible and system instability can arise if the force update rate drops below the range of stable values.

Gregory et al. [GME+00] presented a 6-DoF haptic rendering system that combined collision detection based on convex decomposition of polygonal models [EL01] (see Section 3.3.4), predictive estimation of penetration depth, and force and torque interpolation. They were able to handle interactively dynamic scenes with several convex objects, as well as pairs of nonconvex objects with a few hundred triangles and rather restricted motion (see an example in Fig. 2.3).

Nelson et al. [NJC99] introduced a technique for haptic interaction between pairs of parametric surfaces. Their technique tracks contact points that realize locally maximum penetration depth during surface interpenetration. Tracking contact points, instead of recomputing

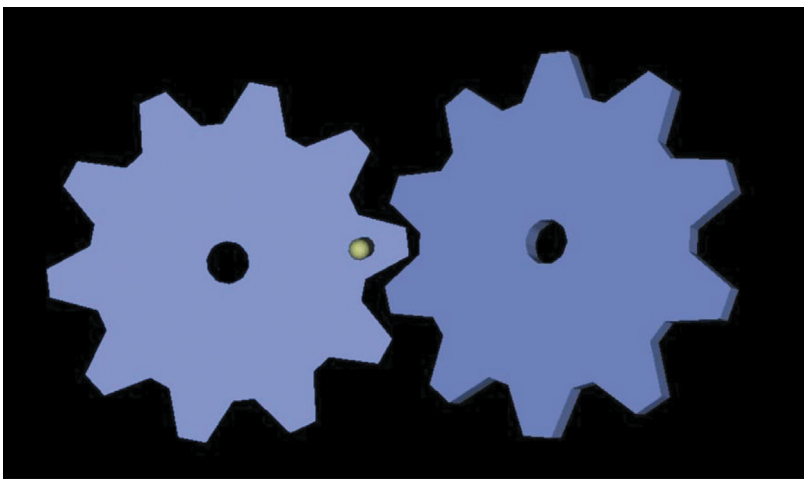

FIGURE 2.3: *3D gear interaction.* Example of direct rendering by Gregory et al. [GME+00].

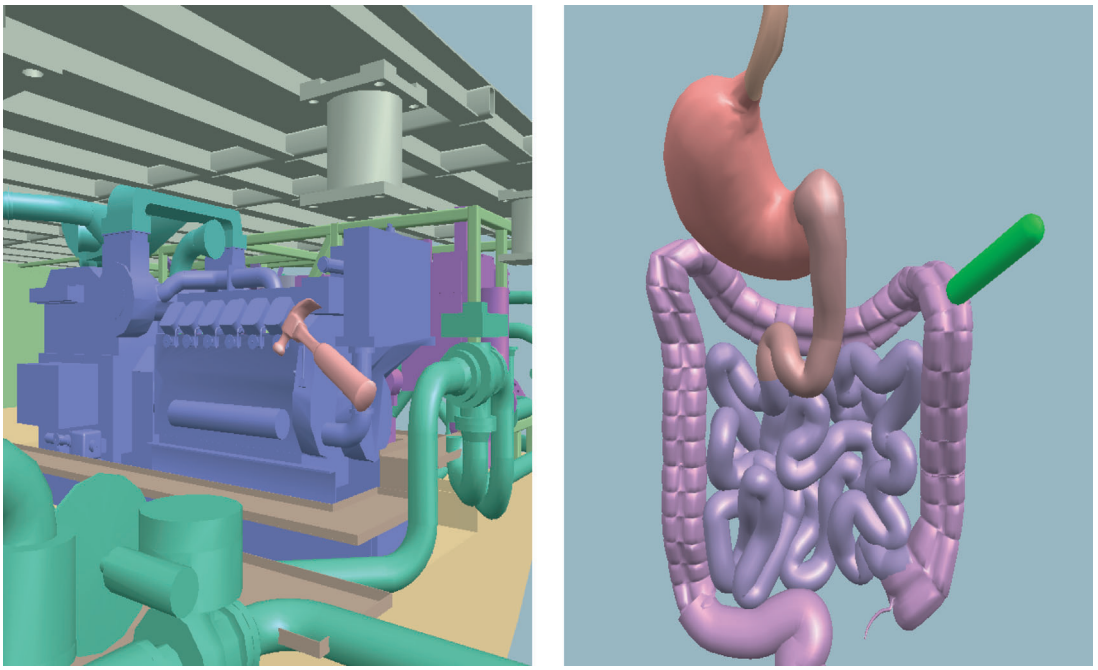

FIGURE 2.4: *Haptic rendering with localized contact computations.* Two contact scenarios displayed with the approach by Kim et al. [KOLM03] (©MIT Press, 2003).

them for every frame, ensures smooth penetration values, which are used for penalty-based force feedback. The contact points are solved in parametric space and they are defined as those pairs of points for which their difference vector is collinear with surface normals.

Johnson and Willemsen [JW03] suggested a technique for polygonal models that defines contact points as those that satisfy a local minimum-distance criterion. Johnson and Willemsen exploited this definition in a fast collision-culling algorithm, using spatialized normal cone hierarchies [JC01] (see Section 3.3.5). They later integrated an incremental tracking algorithm that produces approximate but fast collision-detection updates [JW04] (see Section 2.4), as well as a penetration depth estimation algorithm [JWC05]. Examples of the scenes rendered with their approach are shown in Fig. 2.5.

Recently, Kim et al. [KOLM03] exploited convex decomposition for collision detection and incorporated fast, incremental, localized computation of per-contact penetration depth [KLM02a] (see Section 3.4.3). In order to improve stability and eliminate the influence of triangulation on the description of the contact manifold, they introduced the contact clustering technique described in Section 2.1.3.2. Their system was able to interactively handle pairs of models with nearly one hundred convex pieces each, as the ones shown in Fig. 2.4.

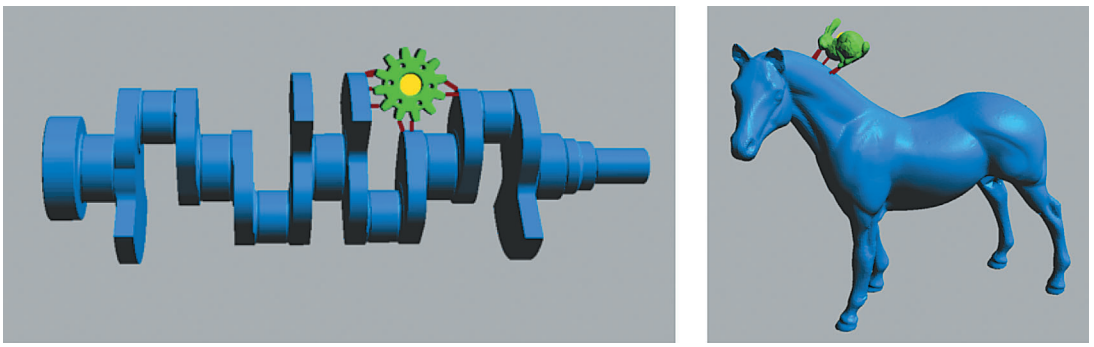

FIGURE 2.5: *Combination of spatialized normal cone search and local descent.* Images courtesy of Johnson et al. [JW03, JW04, JWC05], University of Utah (©2005 IEEE).

2.3 SIMULATION-BASED RENDERING

This category groups together methods where force and torque feedback are generated using a rigid-body simulation of the virtual tool as a fundamental building block. As introduced in Section 1.2.1, user actions exert virtual force and torque on the virtual tool and these are combined with environment forces to produce the resulting motion, which is then used for synthesizing haptic feedback.

There is a diverse set of simulation-based rendering methodologies, but we will review two commonly used approaches characterized by their control strategies. The *virtual coupling architecture* (see Fig. 2.6) is associated with impedance control. The motion of the haptic device is converted to a force acting on the virtual tool by setting a viscoelastic coupling in between [CSB95] and the same force is used as the command for a force control loop. Fig. 2.8 depicts the concept of virtual coupling. In an *admittance control architecture* (see Fig. 2.7), user's actions are directly interpreted as input forces and the motion of the virtual tool is employed as the command signal for a position controller that drives the haptic device. Other more elaborate simulation-based architectures are also possible, such as the four-channel architecture based on teleoperation control designed by Sirouspour et al. [SDS+00].

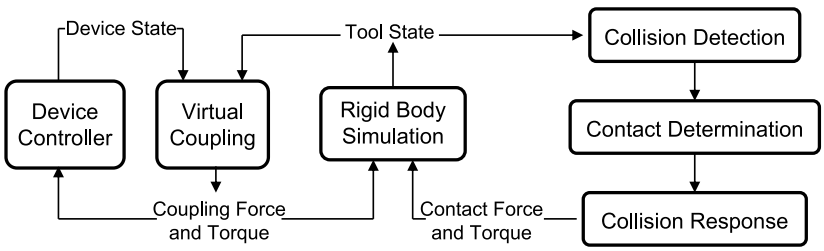

FIGURE 2.6: Virtual coupling architecture.

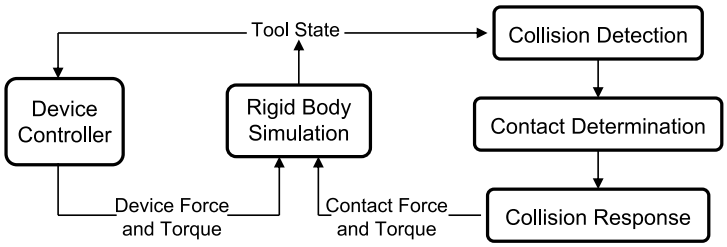

FIGURE 2.7: Admittance architecture.

The main advantage of simulation-based architectures is that the design of stable rendering can be largely simplified. As explained in Section 1.4.3, with virtual coupling stability is guaranteed by implementing a passive simulation of the virtual tool and appropriately tuning the parameters of the viscoelastic coupling. Another important advantage of simulation-based architectures is that object interpenetrations are typically much smaller than with direct rendering architectures. The reason is that in the latter case the contact impedance between the tool and the environment is seriously limited, in order to achieve stable rendering, but this limitation does not exist in simulation-based architectures.

Among the possible disadvantages of simulation-based architectures, perhaps the most notorious one is often the noticeable filtering effects introduced by virtual coupling. This undesirable filtering effect may be due to low update rates on the simulation of the virtual tool, or large tool mass values. Otaduy and Lin [OL05] demonstrated that implicit integration of

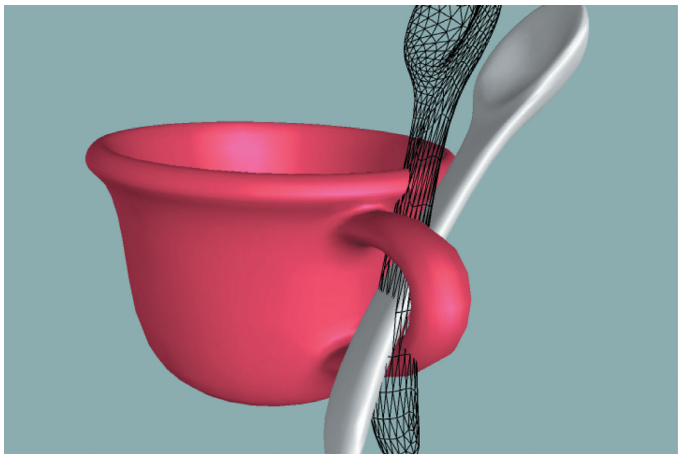

FIGURE 2.8: *Manipulation through virtual coupling.* As the spoon is constrained inside the handle of the cup, the contact force and torque are transmitted through a virtual coupling. A wireframe image of the spoon represents the actual configuration of the haptic device [OL05] (©2005 IEEE).

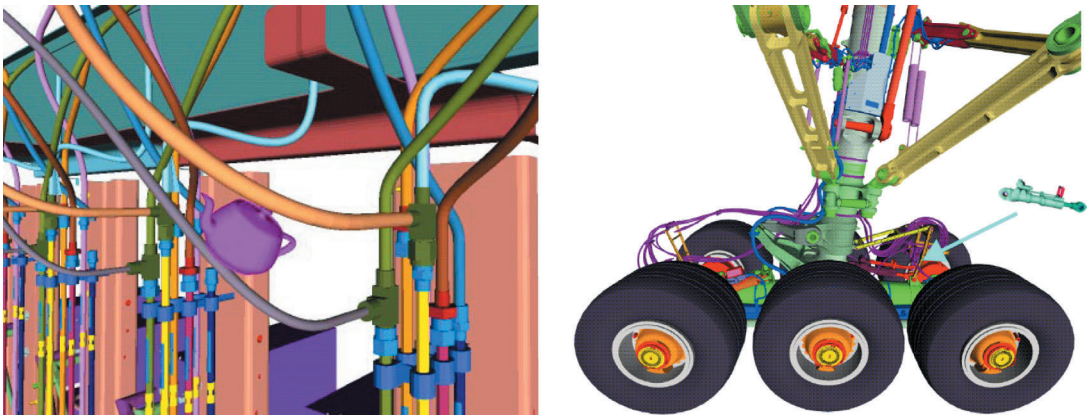

FIGURE 2.9: *Haptic rendering based on voxelization and point-sampling.* On the left, scene rendered with the original approach by McNeely et al. [MPT99], and on the right, scene rendered with their improved version [MPT06] Images courtesy of Boeing (left, ©ACM 1999).

rigid-body simulation of the virtual tool enables low mass values and thus reduces the filtering effect, enhancing the transparency of the haptic display.

Many different 6-DoF haptic rendering algorithms rely on a simulation-based architecture and they differ in the approaches used for collision detection and response. Although some researchers have employed impulse-based [CC97, CSC05] or constraint-based [Ber99, RK00] collision response, penalty-based methods have probably deserved more attention, due to their low computational cost, as discussed in Section 2.1.2.

Recently, constraint-based approaches have regained popularity [ORC06], as part of multirate architectures that reduce the negative effects of complex contact configurations on the update rate (see Section 2.4 for a more detailed discussion).

McNeely et al. [MPT99] presented a system for 6-DoF haptic rendering (see Fig. 2.9) that employs a discrete collision detection approach (see Section 3.5.2) and virtual coupling. The system is intended for assembly and maintenance planning applications. This system has been integrated in a commercial product, VPS, distributed by Boeing. McNeely and his colleagues introduced additional features in order to alleviate some of the limitations. In the original approach, objects were voxelized only on their surface. Recently, McNeely et al. [MPT06] have improved their original approach by employing volumetric distance fields and exploiting geometric awareness and temporal coherence to speed up the computations (more details are given in Section 3.5.2). McNeely et al. [MPT99] also proposed precontact braking forces, similar to the braking impulses suggested by Salcudean [SV94], for reducing the contact velocity of the virtual tool thereby preventing deep penetrations. The existence of multiple contact points produces high stiffness values that can destabilize the simulation of rigid-body dynamics.

Averaging the effects of the different contact points before contact forces are applied to the virtual tool, as described in Section 2.1.3, thus limits the total stiffness and increases the stability of the simulation. The locality of the force model induces force discontinuities when contact points traverse voxel boundaries, but McNeely et al. pointed out that force discontinuities are somewhat filtered by the virtual coupling. Renz et al. [RPP+01] modified McNeely's local force model to ensure continuity of the surface across voxel boundaries, but incurring more expensive force computation.

Using the same voxelization and point-sampling approach for collision detection, Wan and McNeely [WM03] proposed a different solution for computing the position of the virtual tool. The early approach by McNeely et al. [MPT99] computed object dynamics by explicit integration of Newton–Euler equations. Instead, Wan and McNeely [WM03] presented a purely geometric solution that eliminates the instability problems that can arise due to high contact stiffness. Their algorithm formulates linear approximations of the coupling and contact force and torque in the space of translations and rotations of the virtual tool. The state of the tool is computed at every frame by solving for the position of quasi-static equilibrium. Deep penetrations are avoided by formulating the coupling force as a nonlinear spring.

2.4 MULTIRATE METHODOLOGIES

As introduced in Section 1.4.4, multirate approximation techniques have often been used in haptic rendering for separating the execution of collision detection and the synthesis of force feedback into different threads. Such multirate techniques have also been applied in 6-DoF haptic rendering.

By using a linearized contact model as shown in Fig. 2.10, Otaduy and Lin [OL05] separated the haptic rendering process into two threads: a haptic thread that performs the

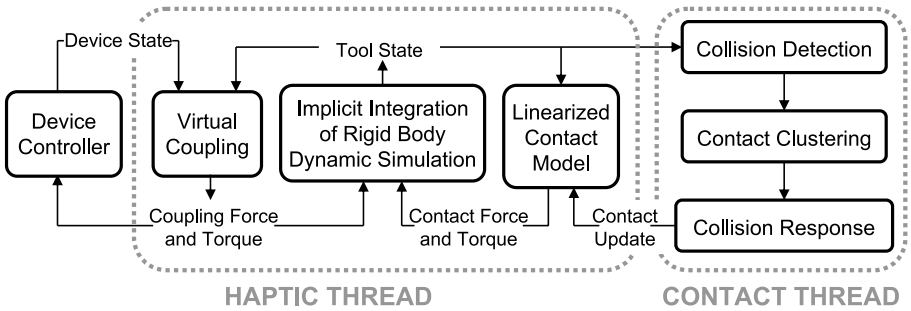

FIGURE 2.10: *Multirate rendering architecture with a linearized contact model.* A haptic thread runs at force update rates simulating the dynamics of the virtual tool and computing force feedback, while a contact thread runs asynchronously and updates contact forces [OL05] (©2005 IEEE).

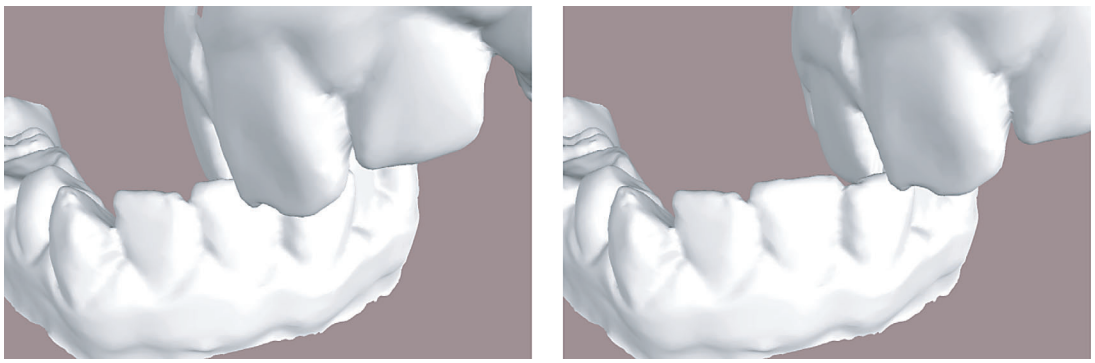

FIGURE 2.11: *Virtual interaction using a linearized contact model.* Dexterous interaction of an upper jaw (47,339 triangles) being moved over a lower jaw (40,180 triangles) using the method by Otaduy and Lin [OL05] (©2005 IEEE).

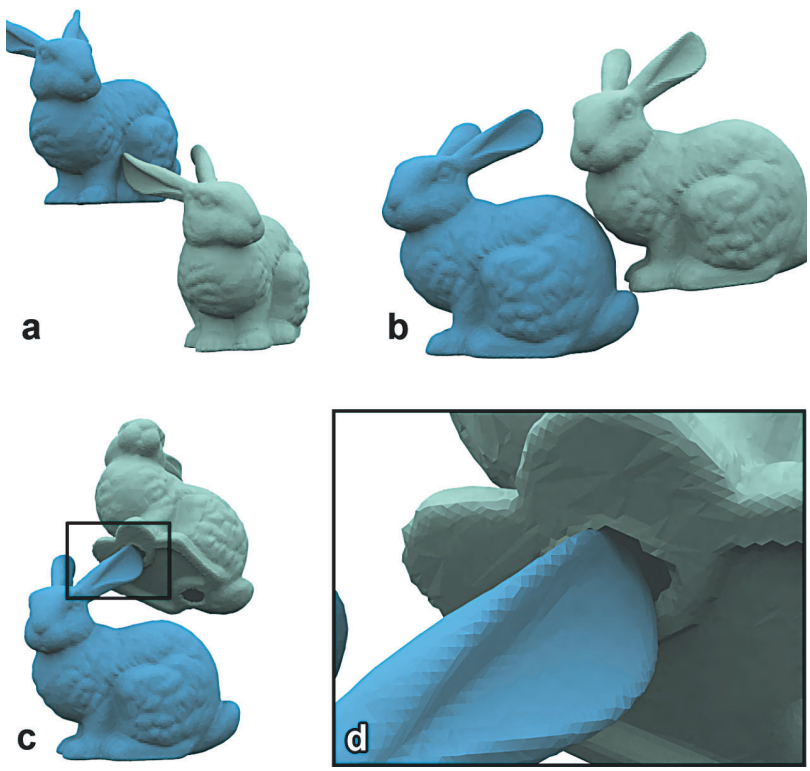

FIGURE 2.12: *6-DoF god-object method.* Images courtesy of Ortega et al. [ORC06], i3D-INRIA-GRAVIR and PSA Peugeot-Citroën (©2006 IEEE).

rigid-body dynamic simulation of the virtual tool and a contact thread that executes collision detection and response. The linearized contact model decouples the process of collision detection from the simulation of rigid-body dynamics, while still enabling a very fast update of collision forces in the simulation. In this way, collision detection is less a bottleneck for the simulation and the synthesis of feedback force and torque. The linearization of contact forces is also efficiently employed for performing implicit integration of tool dynamics (see Section 2.1.1), and this enhances the transparency of the rendering. Previous approaches that exploit multirate architectures with virtual coupling tend to simulate the tool dynamics at the same rate as collision detection, performing only the synthesis of force feedback at a high update rate. As demonstrated by Otaduy and Lin [OL05], the range of stable impedances that can be rendered is notably larger by decoupling the numerical integration from collision detection (see Fig. 2.11). Furthermore, the use of the perceptually driven multiresolution collision detection techniques (see Section 3.6) maximizes the update rate of collision detection.

Johnson and Willemsen [JW04] suggested a different approach for fast updates of collision response. They combined an approximate but fast, incremental contact-point-tracking algorithm with a previous technique for exact but slower collision detection [JW03]. The incremental collision-detection algorithm produces contact information for fast force feedback updates. The exact algorithm is executed asynchronously and after every iteration it updates the initial conditions for the incremental updates. This algorithm handles models with thousands of triangles at interactive rates, although the contact information may suffer some discontinuities if the exact update rate is too slow.

Ortega et al. [ORC06] followed the traditional approach of simulating the dynamics of the virtual tool in a slow thread using constraint-based rigid-body simulation [Ber99], but they introduced a new method for synthesizing feedback force and torque at high update rates. At every iteration of the force control loop, the collision-free acceleration of the virtual tool is constrained based on the active set of constraints and the feedback force (and torque) is synthesized based on the difference between the constrained and unconstrained accelerations. Compared with the virtual coupling approach, the constraint-based coupling of Ortega et al. enables fully transparent rendering during free-space motion and feedback forces that are fully orthogonal to the constraints during contact. The use of continuous collision detection and constraint-based simulation techniques may potentially induce a slow update of the constraints, but the algorithm has been successfully tested on complex objects like the ones shown in Fig. 2.12.

CHAPTER 3

Collision Detection Methods

Collision detection has received much attention in robotics, computational geometry, and computer graphics. Some researchers have investigated the problem of interference detection as a mechanism for indicating whether object configurations are valid or not. Others have tackled the problems of computing separation or penetration distances, with the objective of applying collision response in simulated environments.

We begin this chapter with a distinction of the broad and narrow phases of collision detection [Hub94]. The vast majority of the narrow-phase algorithms used in practice proceed in two steps: first they cull large portions of the objects that are not in close proximity, using spatial partitioning, hierarchical techniques, or visibility-based properties, and then perform primitive-level tests. Among the different existing collision detection algorithms, in this chapter we describe those that have been applied in practice to the problem of 6-DoF haptic rendering: oriented bounding-box trees [GLM96], hierarchies of convex hulls [EL01], spatialized normal cone hierarchies [JC01], and voxelization methods [MPT99]. We also cover briefly the topic of continuous collision detection and the use of graphics processors for collision detection. For more information on collision detection, please refer to surveys on the topic [LG98, KHM+98, LM04].

3.1 BROAD VERSUS NARROW PHASE

The operation of collision detection can be decomposed into two steps: the *broad phase*, where potentially colliding objects are identified, and the *narrow phase*, where colliding primitives are detected [Hub94]. This chapter focuses on the narrow phase of collision detection, but here we briefly discuss the broad phase.

Given n dynamic objects, the cost of detecting potentially colliding objects is $O(n^2)$ in the worst case. In the cases where the size of the output is asymptotically smaller, this cost can be reduced by quickly pruning pairs of objects that do not collide. For example, the visibility-based algorithm CULLIDE [GRLM03] can prune in linear-time objects that do not collide with any other object.

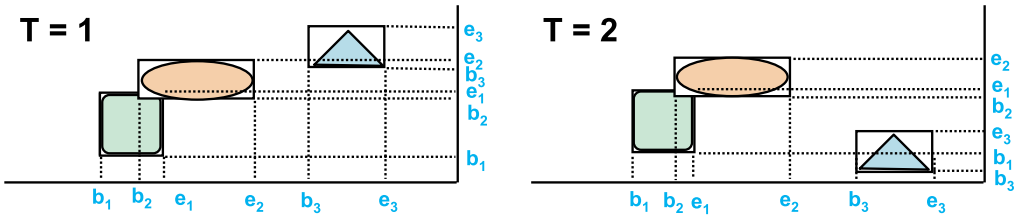

FIGURE 3.1: *Sweep-and-prune.* Pairs of objects are pruned if their projections onto the Cartesian axes do not overlap.

Perhaps the most commonly used algorithm for broad-phase collision detection is *sweep-and-prune* [CLMP95], which detects pairs of potentially colliding objects in 3D by projection to the Cartesian axes and sorting operations.

Fig. 3.1 shows two examples of 2D scenes where the axis-aligned bounding boxes of the objects are projected to the Cartesian axes for 1D overlap tests. The sweep-and-prune algorithm proceeds along the following steps:

1. Compute the axis-aligned bounding box (fixed versus dynamic) for each object.
2. Dimension reduction by projecting boxes onto each x-, y-, z-axis.
3. Sort the endpoints and find overlapping intervals.
4. Possible collision—only if projected intervals overlap in all three dimensions.

Another way of reducing the cost of broad-phase collision detection is to use a scheduling scheme [Lin93]. When the velocity and acceleration of all objects are known at each step, "critical events" to be processed can be prioritized using a heap. Each object pair is tagged with the estimated time to the next collision, and pairs of objects are processed accordingly. The heap is updated when a collision occurs. Given an upper bound on the relative acceleration between any two points on any pair of objects, $a_{\max}$, relative linear acceleration, a_{lin}, relative rotational acceleration, α, relative linear velocity, v_{lin}, relative rotational velocity, ω, the difference between the centers of mass of two bodies, r, and the initial separation for two given objects, d, the time of collision t_c can be estimated as

$$t_c = \left(\sqrt{v_i^2 + 2a_{\max}d} - v_i\right)/a_{\max}, \tag{3.1}$$

$$a_{\max} = |a_{\text{lin}} + \alpha \times r + \omega \times \omega \times r|, \tag{3.2}$$

$$v_i = |v_{\text{lin}} + \omega \times r|. \tag{3.3}$$

3.2 PROXIMITY QUERIES BETWEEN CONVEX POLYHEDRA

The property of convexity has been exploited in algorithms with sublinear cost for detecting interference or computing the distance between two polyhedra. Detecting whether two convex polyhedra intersect can be posed as a linear programming problem, searching for the coefficients of a separating plane. Well-known linear programming algorithms [Sei90] can run in expected linear time due to the low dimensionality of the problem.

The separation distance between two polyhedra A and B is equal to the distance from the origin to the Minkowski sum of A and $-B$ [CC86]. This property was exploited by Gilbert et al. [GJK88] in order to design a convex optimization algorithm (known as GJK) for computing the separation distance between convex polyhedra, with linear-time performance in practice. Cameron [Cam97] modified the GJK algorithm to exploit motion coherence in the initialization of the convex optimization at every frame for dynamic problems, achieving nearly constant running-time in practice.

Lin and Canny [LC91, Lin93] designed the *Voronoi marching* algorithm for computing separation distance by tracking the closest features between convex polyhedra. The motivation behind the algorithm is to partition the space into cells such that all points in a cell are closest to only one primitive. The geometric primitives are referred to as *Voronoi sites*, while the set of points closest to a particular site or feature (vertex, edge, or face) is a *Voronoi region*. The set of all Voronoi regions is the *generalized Voronoi diagram*. The algorithm "walks" on the surfaces of the polyhedra until it finds two features that lie on each other's Voronoi regions. By starting the search every iteration at the closest features from the previous time step, the algorithm exploits motion coherence and geometric locality. Voronoi marching runs in nearly constant time per frame, independent of the geometric complexity of the objects.

Mirtich [Mir98] later improved the robustness of this algorithm. Fig. 3.2 shows simple examples of Voronoi marching in 2D and 3D. In the 2D example, P1 and P2 are the closest features of objects A and B. Note that P1 and P2 lie in each other's Voronoi regions.

Given polyhedra A and B with m and n polygons respectively, Dobkin and Kirkpatrick [DK90] proposed an algorithm for interference detection with $O(\log m \log n)$ time complexity that uses hierarchical representations of the polyhedra. Others have also exploited the use of hierarchical convex representations along with temporal coherence in order to accelerate queries in dynamic scenes. Guibas et al. [GHZ99] employ the inner hierarchies suggested by Dobkin and Kirkpatrick, but they perform faster multilevel walking. Ehmann and Lin [EL00] employ a modified version of Dobkin and Kirkpatrick's outer hierarchies, computed using simplification techniques, along with a multilevel implementation of Lin and Canny's Voronoi marching [LC91].

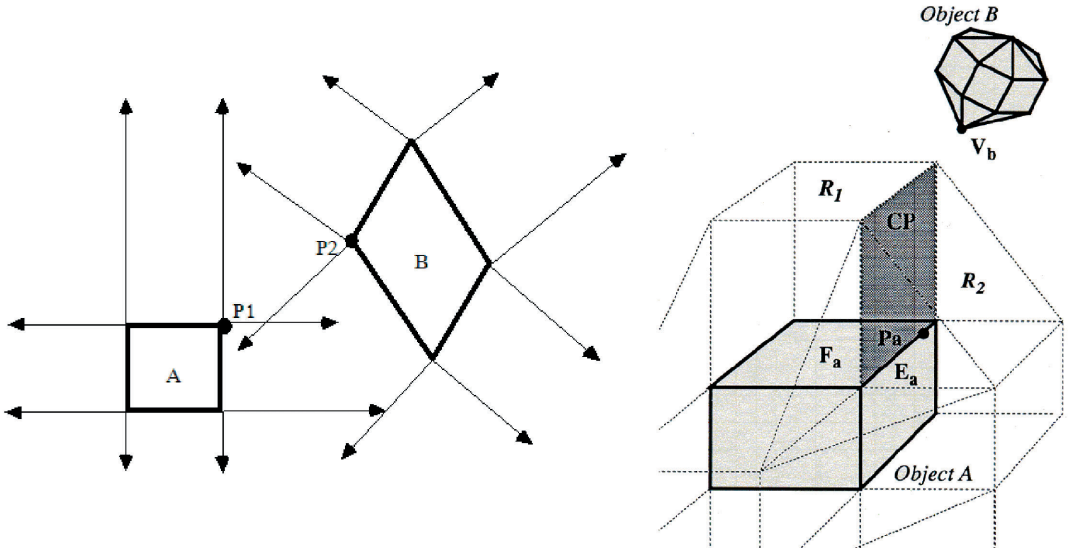

FIGURE 3.2: *Voronoi marching.* Left: 2D example of two features, P1 and P2, in each other's Voronoi regions. Right: Voronoi marching in 3D.

3.3 HIERARCHICAL COLLISION DETECTION

The algorithms for collision detection between convex polyhedra are not directly applicable to nonconvex polyhedra or models described as polygon soups. Brute force checking of all triangle pairs, however, is usually unnecessary. Collision detection between general models achieves large speedups by using hierarchical culling or spatial partitioning techniques that significantly reduce the number of primitive-level tests (e.g., triangle–triangle intersection tests). Over the last decade, bounding volume hierarchies (BVH) have proven successful in accelerating collision culling for dynamic scenes of rigid bodies. For an extensive description and analysis on the use of BVHs for collision detection, please refer to Gottschalk's PhD dissertation [Got00].

3.3.1 Bounding Volume Hierarchies

Assuming that an object is described by a set of triangles T, a BVH is a tree of BVs, where each BV C_i bounds a cluster of triangles $T_i \in T$. The clusters bounded by the children of C_i constitute a partition of T_i. The effectiveness of a BVH is conditioned by ensuring that the branching factor of the tree is O(1) and that the size of the leaf clusters is also O(1). Often, the leaf BVs bound only one triangle. A BVH may be created in a top-down manner, by successive partitioning of clusters, or in a bottom-up manner, by using merging operations.

In order to perform interference detection using BVHs, two objects are queried by recursively traversing their BVHs in tandem, as shown in Fig. 3.3. Each recursive step tests whether

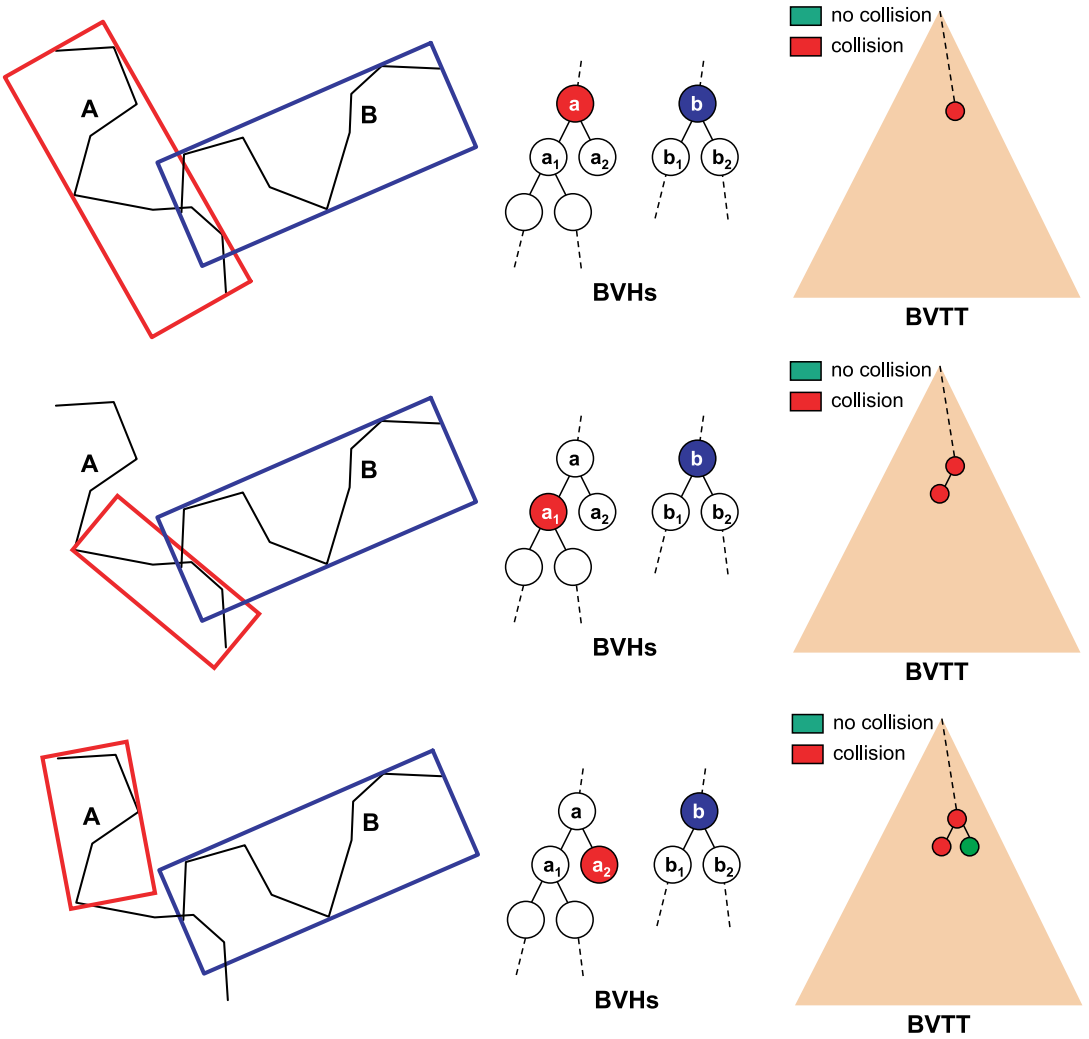

FIGURE 3.3: *BVH traversal.* From left to right, test of BVs in object space, schematic representation of the BVHs, and BVTT showing positive and negative tests. The collision test is performed by traversing the BVHs in tandem, and the pairwise BV tests can be represented using the BVTT.

a pair of BVs a and b, one from each hierarchy, overlap. If a and b do not overlap, the recursion branch is terminated. Otherwise, if they overlap, the algorithm is applied recursively to their children. If a and b are both leaf nodes, the triangles within them are tested directly. This process can be generalized to other types of proximity queries as well.

The test between two BVHs can be described by the *bounding volume test tree* (BVTT) [LGLM00], a tree structure that holds the result of the query between two BVs

in each node. In situations with temporal coherence, collision tests can be accelerated by *generalized front tracking* (GFT) [EL01]. GFT caches the front of the BVTT where the result of the queries switches from true to false for initializing the collision query in the next time step. The overall cost of a collision test is proportional to the number of nodes in the front of the BVTT. When large areas of the two objects are in close proximity, a larger portion of the BVTT front is close to the leaves, and it consists of a larger number of nodes. The size of the front also depends on the resolutions with which the objects are modeled; higher resolutions imply a deeper BVTT. To summarize, the cost of a collision query depends on two key factors: the size of the contact area and the resolutions of the models.

3.3.2 Choices of Bounding Volumes

One determining factor in the design of a BVH is the selection of the type of BV. Often there is a tradeoff among the tightness of the BV (and therefore the culling efficiency), the cost of the collision test between two BVs, and the dynamic update of the BV (especially relevant for deformable models). Some of the common BVs, sorted approximately according to increasing query time, are spheres [Qui94, Hub94], axis-aligned bounding boxes (AABB) [BKSS90], oriented bounding boxes (OBB) [GLM96], k-discrete-orientation polytopes (k-DOP) [KHM+98], convex hulls [EL01], and swept sphere volumes (SSV) [LGLM00]. BVHs of rigid bodies can be computed as a preprocessing step, but deformable models require a bottom-up update of the BVs after each deformation.

Recently, James and Pai [JP04] have presented the BD-tree, a variant of the sphere-tree data structure [Qui94] that can be updated in a fast top-down manner, if the deformations are bounded and can be described by a small number of parameters.

3.3.3 OBB Trees

OBBs [GLM96] are one of the most popular choices for collision detection between rigid bodies due to their good tradeoff between bounding tightness and cost of the overlap test. They have also been used in 6-DoF haptic rendering [ORC06].

The overlap test between two OBBs relies on the search of a separating axis. As illustrated in Fig. 3.4, axis L is a separating axis for two OBBs A and B, if and only if the projections of A and B onto L yield two disjoint intervals. Given the distance s between the projections of the centroids of A and B, and half the size of the projected intervals, h_a and h_b, the intervals overlap if and only if $s > h_a + h_b$.

As discovered by Gottschalk et al. [GLM96], the overlap test can be reduced to testing a small finite number of axes. In 2D, four axes are sufficient, and in 3D 15 as defined by the normals of the faces of the OBBs and their pairwise cross-products. Such concise overlap test offers multiple advantages: a small constant number of arithmetic operations per overlap test;

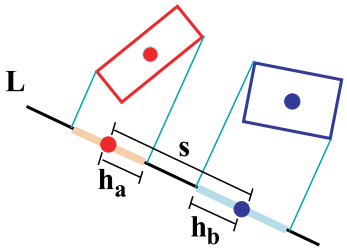

FIGURE 3.4: *Separating axis theorem.* If the projected intervals do not overlap, then the boxes themself do not overlap either.

no special cases for parallel/coincident faces, edges, or vertices; no special cases for degenerate boxes; no conditioning problems; and suitability for microcoding.

The overlap test between two OBBs is, however, more expensive than for other simpler BVs such as AABBs or spheres. The main advantage of OBBs is given by their superior fitting properties over simpler bounding volumes like AABBs and spheres. Fig. 3.5 outlines the construction of an OBB. Given a set of points (e.g., the vertices of the object), the idea is to find a main axis for the OBB such that its volume is minimized. In practice, this can be approximated by projecting the points onto the line and maximizing the variance of the distribution of the projected points [GLM96]. The orientation of the OBB is given by the eigenvectors of the covariance matrix of the original set of points. For polyhedral models with irregular sampling,

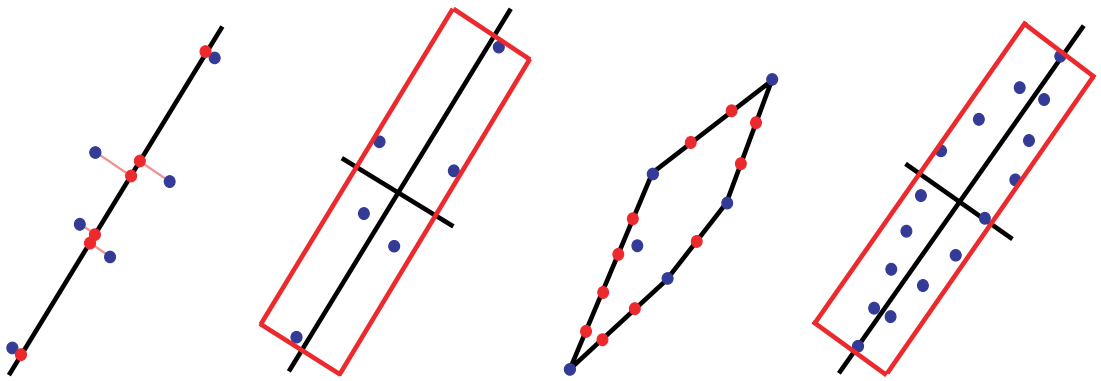

FIGURE 3.5: *Building an OBB (from left to right).* (a) A set of points is projected onto a line, such that the variance of the distribution of the projections is maximized; (b) The directions of the OBB are given by the eigenvectors of the covariance matrix of the original points; (c) For polyhedral models, it is convenient to take the convex hull and regularly sample the facets of the convex hull. (d) In this way the computation of the OBB is more robust and is not influenced by the original sampling.

it is convenient to first compute the convex hull of the model and regularly sample the faces of the convex hull, thus enhancing the robustness of the OBB computation.

3.3.4 Convex Hull Hierarchies—SWIFT++

The SWIFT++ collision detection algorithm by Ehmann and Lin [EL01] is based on BVHs of convex hulls. Ehmann and Lin defined both the process to create the BVHs and the collision queries using convex polyhedra. The algorithm imposes some restrictions on the input models, as they must be orientable 2-manifold polyhedral surfaces, but it offers equally good or better performance than other collision detection algorithms for a variety of proximity queries. Specifically, its performance is as good as that of algorithms based on BVHs of OBBs for intersection queries, but it offers the capability to return additional collision information (i.e., distances, closest points, contact normals), which is very useful for collision response in 6-DoF haptic rendering. The reason behind this additional capability is that the collision queries are performed between convex surface patches of the original objects, rather than using bounding volumes with an offset (e.g., OBBs, AABBS, k-DOPs, or bounding spheres).

As a preprocessing, a convex surface decomposition must be performed on the input objects. Ehmann and Lin extended the convex surface decomposition procedure of Chazelle et al. [CDST97], by setting additional constraints such that the convex volume that encloses a convex patch does not protrude the surface of the object. The convex volumes that enclose the initial patches constitute the leaves of the BVH. Then the hierarchy is constructed by pairwise merging convex patches and constructing the convex hulls of the unions. Fig. 3.6 shows the convex surface decomposition and convex hull hierarchy for a model of a torus. Fig. 3.6(a) shows the torus model, while Fig. 3.6(b) to 3.6(h) depict the different levels of the BVH. Original faces of the torus are colored in green; virtual faces that fill the convex hulls of the convex patches are colored in yellow; and virtual faces that fill convex hulls to construct the hierarchy are colored in red.

The runtime collision detection algorithm of SWIFT++ follows the basic culling strategy of BVHs. The query between two convex hulls is performed using an extension of the Voronoi marching algorithm [LC91]. This allows finding the closest features between convex patches and computing distance information. SWIFT++ also exploits properties of convex hulls to enable early exit in intersection tests. When some convex hulls in the BVHs intersect, there is no need to continue with the recursive BV tests if the intersecting triangles lie on the original surfaces. Finally, SWIFT++ exploits motion coherence by applying generalized front tracking and caching the closest primitives at every pair of tested convex hulls in the previous frame. At the new frame, Voronoi marching starts from the cached primitives, achieving O(1) query time for each pair of convex hulls in most situations.

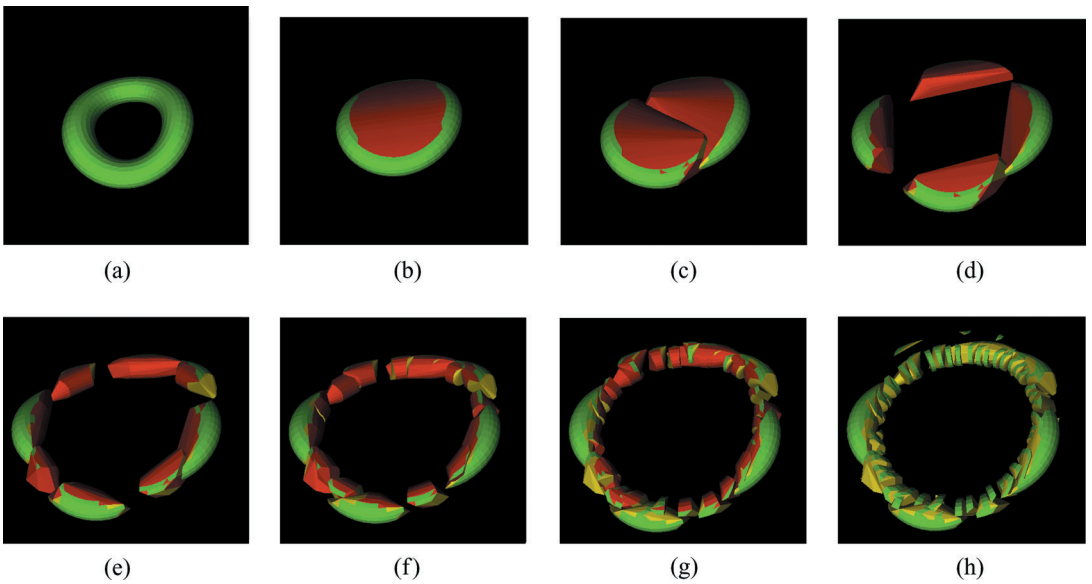

FIGURE 3.6: *Convex decomposition.* (a) Model of a torus. (b)–(h) Different levels of the convex hull hierarchy. Green indicates original faces, yellow indicates virtual faces that enclose the convex hulls of the convex surface decomposition, and red indicates faces used for constructing the convex hull hierarchy [EL01] (©ACM, 2002).

3.3.5 Spatialized Normal Cone Hierarchies

Spatialized normal cone hierarchies [JC01] enable finding the local minimum distances (LMDs) between closed surfaces. They exploit the definition of locally closest points for parametric surfaces [Sny95]. Assuming two parametric surfaces, $F(u, v)$ and $G(s, t)$, the squared Euclidean distance between them can be written as $D^2(u, v, s, t) = \|F(u, v) - G(s, t)\|^2$. Differentiation of this formula yields the following conditions for distance extrema, based on the partial derivatives of the surface functions:

$$\begin{aligned}(F - G) \cdot F_u &= 0,\\ (F - G) \cdot F_v &= 0,\\ (F - G) \cdot G_s &= 0,\\ (F - G) \cdot G_t &= 0.\end{aligned} \tag{3.4}$$

Intuitively, these equations imply that, at distance extrema, the surface normals are collinear to the difference vector between surface points. For polyhedral models, two primitives (vertex, edge, or face) define a LMD if the difference vector between them lies in their normal cones.

Johnson and Cohen [JC01] proposed the spatialized normal cone hierarchies as a data structure for detecting LMDs between polyhedral models in a hierarchical manner. Every

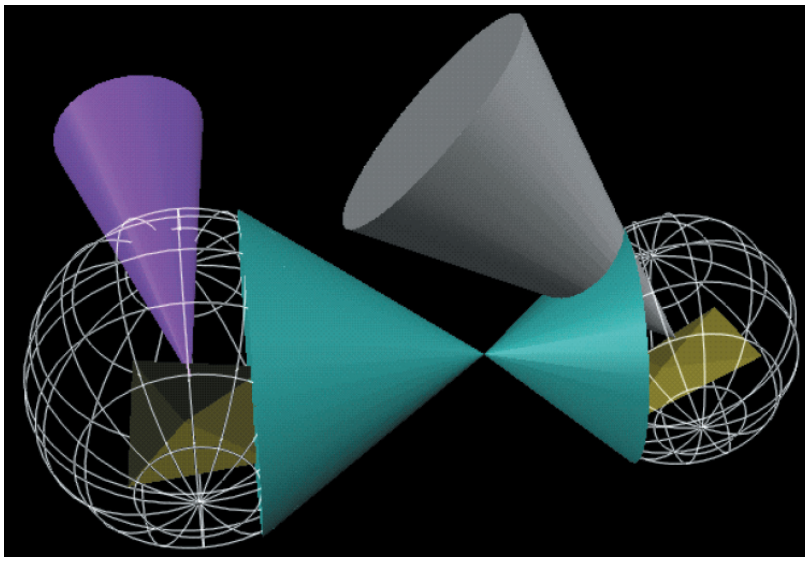

FIGURE 3.7: *Bounding spheres and normal cones.* Schematic view of the overlap test between two spatialized normal cones, using a dual view cone (in green). Image courtesy of Johnson et al. [JC01, JWC05], University of Utah (©2005 IEEE).

primitive is associated with its bounding sphere and its normal cone, and primitives are grouped to define a BVH. Fig. 3.7 shows the spatialized normal cone for two models.

The LMD test for two spatialized normal cones is based on the definition of a *dual view cone*, as depicted in Fig. 3.7. The dual view cone spans all the possible vectors between the two bounding spheres. One needs to test if the normal cones overlap with the dual view cone. If they do, the recursive process continues as usual with BVHs until the leaves are reached.

Johnson and Willemsen [JW03] applied spatialized normal cone hierarchies to 6-DoF haptic rendering by computing penalty-based forces (see Section 2.1.2) between points defining LMDs. In practice, they used a small cutoff distance to prune minimum distance pairs that were too far away, as indicated in Fig. 3.8. This can be easily handled in a hierarchical manner by testing for distances between pairs of bounding spheres.

Later, spatialized normal cone hierarchies have been extended for performing local LMD search [JW04]. As explained in Section 2.4, multirate haptic rendering provides a fast update of feedback forces, while exact collision detection may be running at a lower update rate. The general idea is to perform local walks on surface primitives based on distance gradient descent. These local updates are only accurate in convex regions of the objects, and they reduce to the local walk of Voronoi marching [LC91] (see also Section 3.2).

Johnson and Willemsen [JWC05] also proposed an extension of spatialized normal cone hierarchies for estimating penetration depth [JWC05]. Instead of using the traditional definition

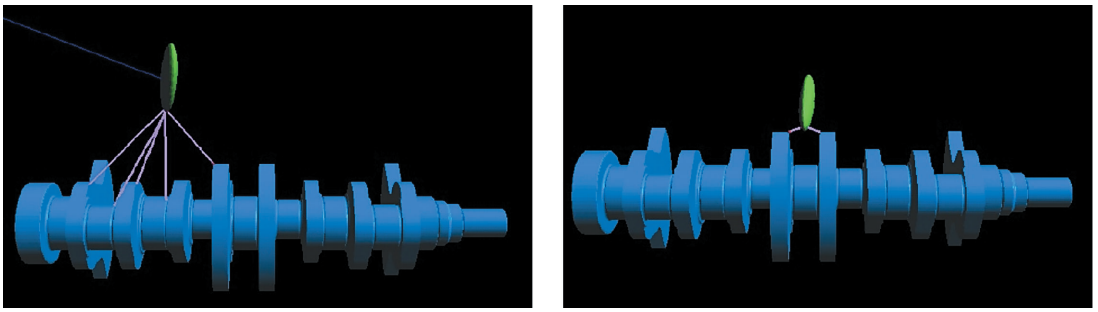

FIGURE 3.8: *Local minimum distances.* On the left, several LMDs detected by the spatialized normal cone hierarchies. On the right, LMDs up to a cutoff distance. Images courtesy of Johnson et al. [JW03, JWC05], University of Utah (©2005 IEEE).

based on minimum separation distance (see Section 3.4), they defined contact points for penetrating situations as local distance extrema. For higher levels of the hierarchy, the colinearity test between normal cones holds, but at primitive level multiple normal ranges may overlap in the interior of the models. Johnson and Willemsen suggested approximating primitive-level information by using closest points instead of distance extrema, under the assumption that models are sufficiently highly sampled such that the error is small.

For more information on the application of spatialized normal cone hierarchies to haptic rendering, a complete summary is given in [JWC05].

3.3.6 Continuous Collision Detection

Continuous collision detection refers to a temporal formulation of the collision detection problem. The collision query attempts to find intersecting triangles and the time of intersection. Redon et al. [RKC02] proposed an algorithm that assumes an arbitrary interframe rigid motion and incorporates the temporal dimension in OBB-trees using interval arithmetic. Continuous collision detection enables constraint-based simulations without backtracking operations used for computing the time of first collision, and it has recently been applied to 6-DoF haptic rendering [ORC06]. Continuous collision detection performs well in situations with moderate motion relative to the resolution of the models, but produces a ghost viscosity effect in the rendering during fast tangential motion in concave regions.

3.4 PENETRATION DEPTH

The computation of penetration depth deserves a separate section in the topic of collision detection. Penalty-based methods, often used in haptic rendering and described in Section 2.1.2, rely on object interpenetration and the depth of penetration for resolving collisions. After

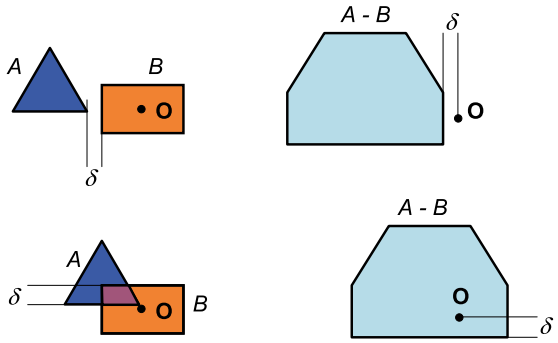

FIGURE 3.9: *Separation distance versus penetration depth.* Top left: separation distance δ between objects A and B in object space. Top right: separation distance in configuration space, shown as the distance from the origin to the boundary of the Minkowski difference. Bottom left: penetration depth δ between objects A and B in object space. Bottom right: penetration depth in object space, shown as the distance from the origin to the boundary of the Minkowski difference. Note that in the penetrating case, the origin of the configuration space lies inside the Minkowski difference.

giving some important definitions, we summarize several approaches for the computation of penetration depth, focusing on the DEEP algorithm by Kim et al. [KLM02a], as this algorithm is exploited in the multiresolution collision detection method described later in Section 3.6.1.

3.4.1 Definitions

The penetration depth between two intersecting polyhedra A and B is defined as the minimum translational distance required for separating them. For intersecting polyhedra, the origin is contained in the Minkowski sum of A and $-B$, and the penetration depth is equal to the minimum distance from the origin to the surface of the Minkowski sum, as shown in Fig. 3.9. The computation of penetration depth can be $\Omega(m^3n^3)$ for general polyhedra [DHKS93].

The Minkowski sum computes the convolution of two shapes A and B as the sum $A + B = \mathbf{p}_a + \mathbf{p}_b | \mathbf{p}_a \in A, \mathbf{p}_b \in B$. The Minkowski difference simply refers to the Minkowski sum where B is negated and then $A + (-B)$ is computed.

3.4.2 Summary of Approaches

Many researchers have restricted the computation of penetration depth to convex polyhedra. In computational geometry, Dobkin et al. [DHKS93] presented an algorithm for computing directional penetration depth, while Agarwal et al. [AGHP+00] introduced a randomized algorithm for computing the penetration depth between convex polyhedra. Cameron [Cam97] extended the GJK algorithm [GJK88] to compute bounds of the penetration depth, and van den Bergen [van01] furthered his work. Kim et al. [KLM02a] presented DEEP, an algorithm

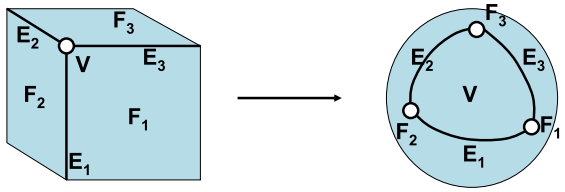

FIGURE 3.10: *Gauss map.* Left: face F, edge E, and vertex V primitives of a cube. Right: the Gauss map of the same cube.

that computes a locally optimal solution of the penetration depth by walking on the surface of the Minkowski sum. This approach is described in detail in the next section.

The computation of penetration depth between general polyhedra presents a very high computational cost, and few practical approaches are known. Kim et al. [KLM02b] presented an algorithm that decomposes the polyhedra into convex patches, computes the Minkowski sums of pairwise patches, and then uses an image-based technique in order to find the minimum distance from the origin to the surface of the Minkowski sums.

3.4.3 Dual-Space Expansion for Convex Polyhedra—DEEP

The DEEP algorithm by Kim et al. [KLM02a] aims at computing the penetration depth between two convex polytopes. It exploits the definition of penetration depth as the minimum distance from the origin to the boundary of the Minkowski sum (or configuration space obstacle, CSO) of two objects A and $-B$.

DEEP exploits the fact that the Minkowski sum of two convex polyhedra can be computed from the arrangement of their Gauss maps. A Gauss map relates a feature by its surface normal in 3D to a location on the surface of a unit sphere, as shown by the example in Fig. 3.10. A face of a polyhedron maps to a point in the Gauss map, an edge maps to a great arc, and a vertex produces a region bounded by great arcs.

In practice, DEEP implicitly computes the surface of the Minkowski sum by constructing local Gauss maps of the objects. Fig. 3.11 provides an overview of the computation of the Minkowski sum from the Gauss maps. The Gauss maps of the convex polytopes to be tested are computed as a preprocess. During runtime, by using a pair of initialization features, the Gauss map of one object is transformed to the local reference system of the other object. The Gauss maps are projected onto a plane, and the arrangement is implicitly computed. The features that realize the penetration depth also define a penetration direction, which corresponds to a certain direction in the intersected Gauss maps. The problem of finding the penetration features reduces to checking distances between vertex-face or edge–edge features that overlap in the Gauss map, as these are the features defining the boundary of the Minkowski sum.

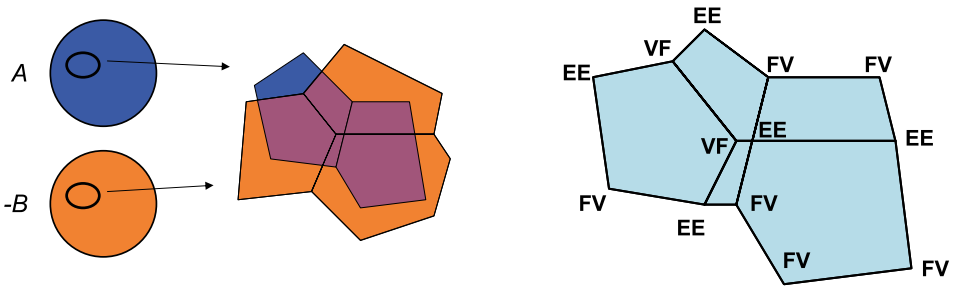

FIGURE 3.11: *Minkowski sum from Gauss map.* Left: Gauss maps of the intersected convex pieces. Middle: overlaid and projected Gauss maps. Right: overlay region that implicitly defines the Minkowski difference. FV, VF, and EE pairs correspond to the vertices of the Minkowski sum [KLM02a].

Given a pair of features that define a vertex of the Minkowski sum, DEEP proceeds by walking to neighboring feature pairs that minimize the penetration depth, until a local minimum is reached. The distance from the origin to the boundary of the Minkowski sum of two convex polytopes may present several local minima, but in practice DEEP reaches the extreme minimum if appropriate initialization features are given. In situations with high motion coherence, the penetration features from the previous frame are usually good initialization features.

Kim et al. [KLM02a, KLM04] performed evaluation tests on DEEP, and found that the query time in situations with high motion coherence is almost constant and the performance is better than previous algorithms [van01], both in terms of query time and variations in the penetration normals.

DEEP has been integrated with SWIFT++ [EL01] for handling the penetration depth between nonconvex objects. First, SWIFT++ handles the hierarchical test using BVHs of convex hulls, and in case leaf convex hulls intersect DEEP computes the penetration depth and the features that realize the penetration depth. The combination of SWIFT++ and DEEP has also been applied to 6-DoF haptic rendering [KOLM03, OL03b].

3.5 METHODS BASED ON DISCRETIZATION

In this section, we review collision detection methods that employ discrete representations of the objects. In the case of distance fields or voxelization methods, the discretization is performed in 3D, while in visibility-based GPU methods, this discretization is performed in 2D, after projection of the objects to a frame buffer.

3.5.1 Distance Fields

In contrast to the previously seen collision detection methods, which rely on polyhedral representations of the objects, distance fields exploit discrete representations. Given two potentially

colliding objects, A and B, the surface of A is point-sampled, and the points are queried against the distance field of B (or vice versa). Distance fields can be used for computing a value of penetration depth for each surface sample point, and this information is typically used for collision response with penalty-based methods, described in Section 2.1.2.

Fisher and Lin [FL01] applied distance fields to the estimation of penetration depth between deformable bodies. They computed and updated the distance fields using fast marching level-sets. Hoff et al. [HZLM01] computed distance fields with an image-based algorithm implemented on graphics hardware, and Sud et al. [SOM04] improved the performance of distance field computation by incorporating culling and clamping techniques. Distance fields offer very fast proximity queries, but their drawback is high memory consumption. Adaptive distance fields [FPRJ00] or computation of distance fields on narrow surface bands [Mau03, SPG03] considerably reduce the memory requirements, at the cost of increased query times or reduced applicability.

3.5.2 Point Sampling and Voxelization

McNeely et al. [MPT99, MPT06] applied a discrete collision detection approach to 6-DoF haptic rendering. They represented objects by point sampling the virtual tool and voxelizing the rest of the objects in the environment, as shown in Fig. 3.12. Note that only a narrow surface band of the objects was voxelized for the purpose of penalty-based force computation. For every colliding point sample, McNeely et al. suggested estimating the penetration depth using a tangent plane force model. As shown in Fig. 3.13, each point sample stores a normal direction corresponding to the original surface. The penetration depth is estimated by fitting a plane with such normal through the center of the voxel. This collision model requires that deep penetrations do not occur, and McNeely et al. suggested measures for this purpose, as explained earlier in Section 2.3.

A more recent approach by McNeely et al. [MPT06], employs distance fields on larger regions for the purpose of estimating the time-to-collision and thereby prioritizing the collision

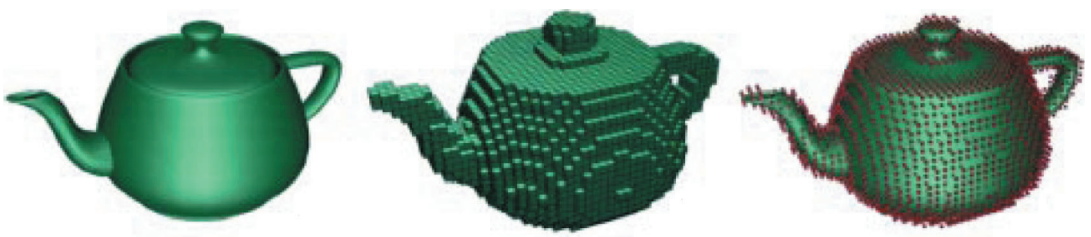

FIGURE 3.12: *Voxelization and point sampling.* Left: original Utah teapot. Middle: voxelized teapot. Right: point-sampled teapot. Images courtesy of McNeely et al. [MPT99], Boeing (©ACM, 1999).

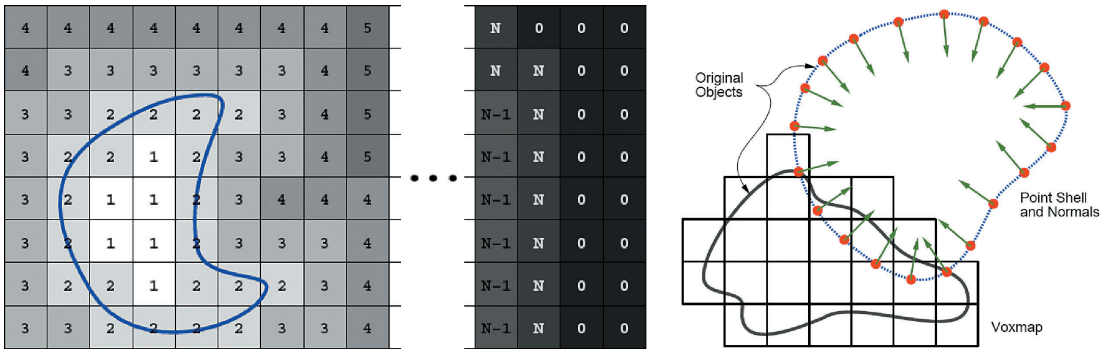

FIGURE 3.13: *Distance fields and contact queries.* Left: distance field in 2D, with $\log_N$ bit encoding. Right: 2D example of contact query with a point-sampled virtual tool and a distance field, following the approach of [MPT99, MPT06]. Images courtesy of Boeing (right, ©ACM, 1999).

queries. McNeely et al. implemented the contact query priority queues with a bit code that indicates the distance from a point to the environment's surface (see Fig. 3.13). A hierarchical representation of the point-sampled virtual tool enables hierarchical culling using this bit code.

3.5.3 Collision Detection Using Graphics Hardware

The processing capability of GPUs is growing at a rate higher than Moore's law, and this trend has stimulated an increasing use of GPUs for general-purpose computation, including collision detection. Rasterization hardware enables high performance of image-based collision detection algorithms. Algorithms for distance field computation discussed above [HZLM01, SOM04] exploit GPU-based rasterization. Others have formulated collision detection queries as visibility problems. Lombardo et al. [LCN99] intersected a complex object against a simpler one using the view frustum and clipping planes and detected intersecting triangles by exploiting OpenGL capabilities. More recently, Govindaraju et al. [GRLM03] have designed an algorithm that performs series of visibility queries and achieves fast culling of nonintersecting primitives in N-body problems with nonrigid motion.

3.6 MULTIRESOLUTION COLLISION DETECTION

Multiresolution collision detection refers to the execution of approximate collision detection queries using adaptive object representations. Hubbard [Hub94] introduced the idea of using sphere trees [Qui94] for multiresolution collision detection, refining the BVHs in a breadth-first manner until the time allocated for collision detection has expired. In a sphere tree, each level of the BVH can be regarded as an implicit approximation of the given mesh, by defining the surface as a union of spheres. Unlike LOD techniques, in which simplification operations

minimize surface deviation, sphere trees add extraneous "bumpiness" to the surface, and this characteristic can adversely affect collision response.

O'Sullivan and Dingliana [OD01] incorporated perceptual parameters into the refinement of sphere trees. Pairs of spheres that test positive for collision are inserted in a priority queue, sorted according to perceptual metrics (e.g., local relative velocity, distance to the viewer, etc.). In this way, the adaptive refinement focuses on areas of the objects where errors are most noticeable.

The use of multiresolution representations for haptic rendering has also been investigated by several researchers. Pai and Reissel [PR97] investigated the use of multiresolution image curves for 2D haptic interaction. El-Sana and Varshney [ESV00] applied LOD techniques to 3-DoF haptic rendering. They created a multiresolution representation of the haptically rendered object as a preprocessing step. At runtime, they represented the object at a high resolution near the probe point and at a low resolution further away. Their approach does not extend naturally to the interaction between two objects, since multiple disjoint contacts can occur simultaneously at widely varying locations without much spatial coherence.

In the remainder of this section, we focus on multiresolution collision detection algorithms for 6-DoF haptic rendering. Otaduy and Lin [OL03b, OL03a] presented *contact levels of detail* (CLODs), dual hierarchical representations designed for perceptually driven multiresolution collision detection in haptic rendering. These representations have been extended and applied to more general rigid-body simulation as well. As discussed in Section 3.3, the cost of collision detection with BVHs can be considered to be linear to the size of the front of the BVTT (in situations with temporal coherence), and this bound depends on the resolution of the models and the size of the contact area. CLODs exploit the perceptual observations discussed in Section 1.3.1 for selecting the resolution of the models adaptively, thus reducing the cost of collision detection.

In this section, we present the multiresolution collision detection techniques using bounding volume hierarchies. However, the principles of the multiresolution collision detection framework and the criteria for adaptively selecting the resolution of the colliding objects can be applied to other methods as well.

3.6.1 Contact Levels of Detail

Efficient multiresolution collision detection depends on two main objectives: (i) create accurate multiresolution representations, and (ii) embed the multiresolution representations in effective bounding volume hierarchies. Multiresolution representations are often created by decimating the given polyhedral models, but difficulties arise when trying to embed these representations in BVHs. Considering each LOD of the given object as one whole model, each LOD would require a distinct BVH for collision detection. This requirement would result in inefficient

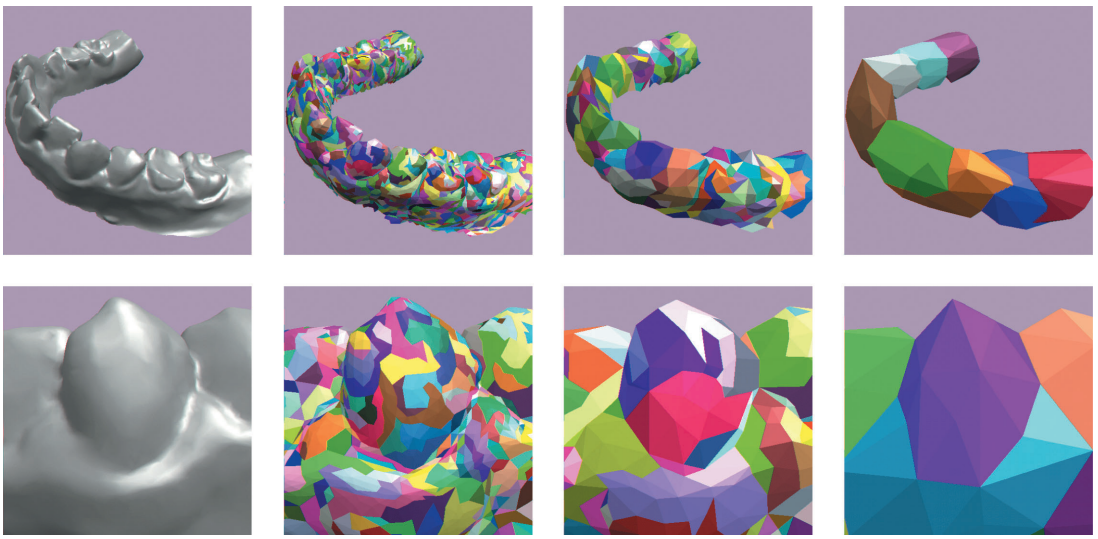

FIGURE 3.14: *CLODs of a lower jaw.* Top: original mesh and some of the CLODs, with the different BVs color coded; Bottom: detailed view of the model. Note the combination of simplification with merging of BVs to construct the BVH [OL03b]. (©ACM, 2003)

collision queries, because the front of the BVTT would have to be updated for the BVH of each LOD.

3.6.1.1 Definition of CLODs

Instead of considering each LOD as one whole model, CLODs constitute a unique dual hierarchical representation, which serves as both a multiresolution representation and a BVH. On one hand, this dual hierarchy constitutes a multiresolution representation built according to haptic error metrics. This feature enables reporting results of contact queries accurate up to some haptic tolerance value. On the other hand, the dual hierarchy constitutes a BVH that enables effective collision detection. Thanks to the dual nature of the data structure, using CLODs in haptic rendering helps to speed up contact queries while maintaining haptic error tolerances. Fig. 3.14 shows several of the CLODs obtained when processing a model of a lower jaw, as well as a more detailed view of the combination of mesh simplification and BVH creation (color-coded).

Assuming that an input model is described as a triangle mesh M_0, the data structure for CLODs consists of the following:

- A sequence of LODs $\{M_0, M_1, \ldots, M_{n-1}\}$, where M_{i+1} is obtained by applying simplification operations to and removing high-resolution geometric detail from M_i.
- For each LOD M_i, a partition of the triangles of M_i into disjoint clusters $\{c_{i,0}, c_{i,1}, \ldots, c_{i,m}\}$.

- For each cluster $c_{i,j}$, a bounding volume $C_{i,j}$.
- A tree T formed by all the BVs of clusters, where BVs of clusters in M_i are children of BVs of clusters in M_{i+1}, and all the BVs except those corresponding to M_0 have at least one child.
- For every BV, $C_{i,j}$, the maximum directed Hausdorff distance $h(C_{i,j})$ from its descendant BVs.

The tree T of BVs, together with the Hausdorff distances, serves as the BVH for culling purposes in collision detection. Directed Hausdorff distances are necessary because, in the definition of CLODs, the set of BVs associated with one particular LOD may not bound the surface of previous LODs. Hausdorff distances are used to perform conservative collision tests.

An additional constraint is added to the data structure, such that the coarsest LOD, M_{n-1}, is partitioned into one single cluster $c_{n-1,0}$. Therefore, the root of the BVH will be the BV of the coarsest LOD. Descending to the next level of the hierarchy will yield the children BVs, whose union encloses the next LOD. At the end of the hierarchy, the leaf BVs will enclose the original surface M_0.

3.6.2 Sensation Preserving Simplification

In a general case, the process of creating the CLODs, depicted in Fig. 3.15, starts by grouping the triangles of the original surface into clusters. The sizes and properties of these clusters depend on the type of BV that is used for the BVH, and will be such that the performance of the collision query between two BVs is optimized. The next step in the creation of CLODs is to compute the BV of each cluster. This initialization is followed by a mesh decimation process along with bottom-up construction of the BVH, carried out by merging clusters and computing the BV of their union.

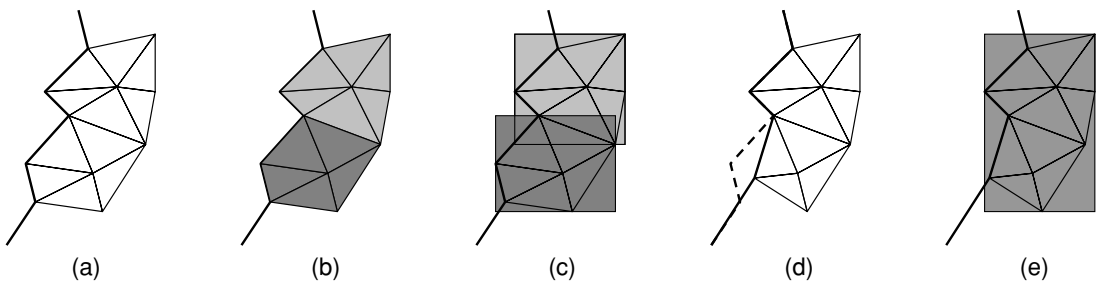

FIGURE 3.15: *Construction of CLODs.* (a) Initial surface; (b) Clusters of triangles; (c) BVs for each cluster; (d) Mesh simplification; (e) BV of the union of clusters after some conditions are met [OL03a] (©ACM, 2003).

Otaduy and Lin [OL03b] implemented CLODs using convex hulls as BVs because, if the clusters are themselves convex surface patches, contact information at triangle level is obtained practically for free when performing the query between BVs. Otaduy and Lin [OL03b] followed the definition of convex surface patches by Ehmann and Lin [EL01], which imposes local and global convexity constraints on the process of creating CLODs: (i) interior edges of convex patches must remain convex after simplification operations are applied, and (ii) the enclosing convex hulls cannot protrude the surface of the object.

The construction of CLODs of convex hulls is initialized by performing a convex surface decomposition of the input object and computing the convex hulls of the resulting convex patches. This is followed by a simplification loop, in which atomic simplification operations are combined with merging of convex hulls. Note that the data structure of CLODs imposes no limitations on the input models, but convex surface decomposition requires the models to be described as 2-manifold, oriented triangle meshes.

After each atomic simplification operation, the union of every pair of neighboring convex patches is tested for convexity. If the union is a valid convex patch itself, the involved patches are merged and the convex hull of the union is computed. All the BVs in LOD M_j that are merged to a common BV $C_{j+1} \in M_{j+1}$ during sensation preserving simplification will have C_{j+1} as their parent in the BVH. A new LOD is output every time the number of convex patches is halved.

Ideally, the process will end with one single convex patch, which serves as the root for the BVH. However, this result is rarely achieved in practice, due to topological and geometric constraints that limit the amount of simplification, and which cannot be removed by local operations. In such cases, the hierarchy is completed by unconstrained pairwise merging of patches [EL01].

3.6.2.1 LOD Resolution and Simplification Priority

In sensation preserving simplification for haptic rendering, the goal is to maximize the resolution at which LODs are generated. As explained in Section 3.6.3, the perceptual error for haptic rendering is measured by taking into account the resolution of the surface detail that is culled away. Multiresolution contact queries will terminate faster as a result of maximizing the resolution at which LODs are generated. From this observation, the edge with highest resolution is selected for collapse at each sensation preserving simplification step. If the edge collapse is successful, the affected edges update their resolutions and priorities, and they are reset as valid for collapse.

The definition of sampling resolution for irregular meshes is inspired by the 1D setting. For a 1D function $F(x)$, the sampling resolution r is the inverse of the distance between two subsequent samples on the real line. This distance can also be interpreted as the projection

of the segment between two samples of the function, v_1 and v_2, onto the average value of the function. The average value is the low-resolution representation of the function itself and can be obtained by low-pass filtering. Extending this idea to irregular meshes, the sampling resolution of an edge $(\mathbf{v}_1, \mathbf{v}_2)$ of the mesh M at resolution r_j, M_j, can be estimated as the inverse of the projected length of the edge onto a low-resolution representation of the mesh, M_{j+1}.

Each LOD M_j is also assigned an associated resolution r_j. This value is the coarsest resolution of all edges collapsed before M_j is generated. Geometrically, it means that the LOD M_j preserves all the detail of the original mesh at resolutions coarser than r_j.

3.6.2.2 Filtered Edge Collapse

The atomic simplification operations in the sensation preserving simplification process must take into account the convexity constraints. For this purpose, Otaduy and Lin [OL03b] suggested the local simplification operation *filtered edge collapse*, inspired by multiresolution analysis and signal processing of meshes. This operation, schematically illustrated in Fig. 3.16, is composed of the following steps:

1. A topological edge collapse. An edge $(\mathbf{v}_1, \mathbf{v}_2)$ is first topologically collapsed to a vertex $\hat{\mathbf{v}}_3$. This step provides down-sampling.
2. An initialization process that sets the position of $\hat{\mathbf{v}}_3$ using quadric error metrics [GH97].
3. Unconstrained relaxation to a position $\tilde{\mathbf{v}}_3$, using Guskov's minimization of second-order divided differences [GSS99].
4. The solution of an optimization problem in order to minimize the distance of the vertex to its unconstrained position, while taking into account the local convexity constraints.
5. A bisection search between the initial position of the vertex and the position that meets the local constraints, in order to find a location where self-intersection constraints and global convexity constraints are also met.

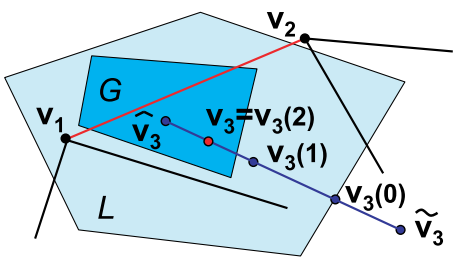

FIGURE 3.16: *Filtered edge collapse with convexity constraints.* The figure illustrates the process of a filtered edge collapse operation. G and L represent feasible regions of global and local constraints respectively. (©ACM, 2003).

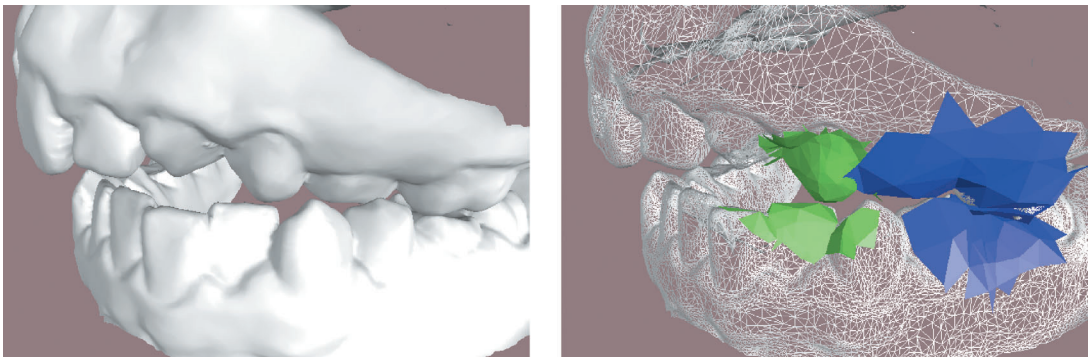

FIGURE 3.17: *Multiresolution collision detection using CLODs.* Left: moving jaws in contact, rendered at their highest resolution. Right: the appropriate object resolution (shown in blue and green) is adaptively selected at each contact location, while the finest resolution is displayed in wireframe [OL03b] (©ACM, 2003).

3.6.3 Multiresolution Contact Queries

Using CLODs, multiresolution collision detection can be implemented by slightly modifying the typical collision-detection procedures based on BVHs. The decision of splitting a node ab of the BVTT is made as a combination of the contact query and a selective refinement query. First, the distance query is performed. If the query returns false, there is no need to descend to the children nodes. If the result of the distance query is true, the query for selective refinement is performed on ab. If the node ab must be refined, the traversal continues with the children of ab in the BVTT. Otherwise, contact information can directly be computed for ab.

Descending to children BVTT nodes involves descending to the children BVs, as occurs in any BVH, but it also involves refining the surface representation, due to the duality of CLODs. Selective refinement of nodes of the BVTT activates varying contact resolutions across the surfaces of the interacting objects, as shown in Fig. 3.17. In other words, every contact is treated independently and its resolution is selected in order to cull away negligible local surface detail.

3.6.3.1 Modification of the Contact Query

A collision detection algorithm based on BVHs must ensure that if a leaf node ab of the BVTT returns true to the contact query, then all its ancestors must return true as well. This is usually achieved by ensuring that the union of the BVs at every level of a BVH fully contains the surface of the object. In CLODs this containment property may not hold, but the correctness of the collision detection can be ensured by modifying the collision distance d_{ab} between two BVs a and b. Given a contact query between objects A and B with distance tolerance d, the distance

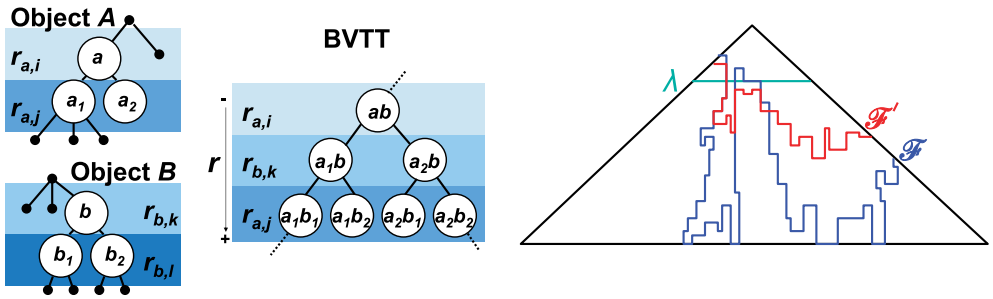

FIGURE 3.18: *CLODs on the BVTT.* Left: BVTT in which levels are sorted according to increasing CLOD resolution. Right: the front of the BVTT for an exact contact query, $\mathbb{F}$, is raised up to the new front $\mathbb{F}'$ using CLODs (©ACM, 2003).

tolerance d_{ab} for the contact query between BVs a and b may be computed as

$$d_{ab} = d + h(a) + h(b), \tag{3.5}$$

where $h(a)$ and $h(b)$ are maximum directed Hausdorff distances from the descendant BVs of a and b to a and b respectively. The Hausdorff distances can be precomputed during the process of sensation preserving simplification.

3.6.3.2 Effects on the Front of the BVTT

Due to the addition of selective refinement, with CLODs the active front of the BVTT, $\mathbb{F}'$, is above the original front $\mathbb{F}$ that separates nodes that test positive to distance queries q from nodes that test negative, as depicted in Fig. 3.18. The front does not need to reach the leaves of the BVTT, as long as the error is smaller than a predefined tolerance. This approach results in a much faster processing of contact queries and, ultimately, it enables 6-DoF haptic rendering of complex objects.

As pointed out in Section 3.6.2, the top levels of the BVHs are obtained by unconstrained pairwise merging of convex patches. These top levels of the BVTT, indicated by the line λ in Fig. 3.18, have no associated metric of resolution and they always test positive for selective refinement.

3.6.3.3 Resolution-Based Ordering of CLODs

When both the contact query and the selective refinement test return true for a node ab of the BVTT, the BV whose children have coarser resolution is split. This splitting policy yields a BVTT in which the levels of the tree are sorted according to their resolution, as shown in Fig. 3.18. A resolution-based ordering of the BVTT is a key factor for maximizing the performance of runtime collision detection, because CLODs with lower resolution and larger error are stored closer to the root of the BVTT. Descending along the BVTT has the effect

of refining the CLODs. The resolution-based ordering of CLODs of two different objects is possible because the definition of resolution described in Section 3.6.2 is an object-independent absolute metric. If objects are scaled, the value of resolution must be scaled accordingly.

3.6.3.4 Selective Refinement and Error Metrics

As discussed in Section 1.3.1, the perceptibility of surface features depends on the ratio of their size and the contact area. This observation leads to the design of error metrics for the selective refinement test. Given a node ab of the BVTT, the idea behind the selective refinement test is to evaluate whether a weighted surface deviation s^*_{ab}, computed based on the size of filtered features and the contact area, is larger than a predefined error tolerance s_0. If s^*_{ab} is above the threshold, the node ab must be refined. Otherwise, the filtered features are considered to be imperceptible.

The weighted surface deviation s^* is computed by averaging over an estimated contact area D the volume estimates ϕ_a and ϕ_b of the filtered features:

$$s^*_{ab} = \frac{\max(\phi_a, \phi_b)}{D},$$
$$D = \max(D_a, D_b), \qquad \phi_a = \frac{s_a}{r_a^2}, \qquad \phi_b = \frac{s_b}{r_b^2}, \tag{3.6}$$

where s_a is the surface deviation from the convex patch bounded by a to the original surface, and r_a is the resolution of the current CLOD. Note that both values can be precomputed during sensation preserving simplification.

Similarly, the online computation of the contact area between a pair of convex patches is too expensive, given the runtime constraint of haptic rendering. Therefore, the contact area D is estimated by selecting the maximum support area of the contact primitives, which can be computed as a preprocess.

Ideally, the surface deviation tolerance s_0 should be a distance defined based on human perceptibility thresholds. However, such metric would be independent of object size and polygon count, and it might result in excessively large, intractable CLOD resolutions. Instead, s_0 can be defined as a metric relative to the size of the interacting objects, under the assumption that the range of motion of the virtual tool covers approximately the space occupied by the objects in the virtual workspace. As a consequence, the required CLOD resolutions are independent of the scale of the objects, and the contact queries run in nearly constant time, as demonstrated by Otaduy and Lin [OL03b]. Based on informal user studies, they estimated values of s_0 between 2.5% and 5% of the radii of the interacting objects. Note that the error metrics, computed for every node in the front of the BVTT, can also be used to prioritize the refinement, enabling time-critical collision detection.

3.6.4 Dynamic Levels of Detail

By definition, CLODs are static LODs, which may lead to discontinuities (i.e, "popping effects") when the active LODs switch. If CLODs are combined with penalty-based collision response, discontinuities can be tackled by interpolating contact information from two LODs. If CLODs are used along with collision response methods that require accurate detection of the time of collision, special treatment is necessary so that switching LODs does not generate inconsistencies or deadlock situations in the time-stepping algorithm.

Yoon et al. [YSLM04] extended the definition of CLODs to handle dynamic LODs, by performing multiresolution collision detection with a cluster hierarchy of progressive meshes [Hop96]. They employed OBBs as BVs, which relax some of the geometric constraints in the construction of CLODs, and are better suited for creating dynamic LODs, but they lose the benefits of CLODs for obtaining contact information at almost no cost.

CHAPTER 4

Haptic Texture Rendering

Rendering of surface texture (i.e., fine geometric features on an object's surface) is an important topic in haptics that has received increasing attention. The intrinsic surface property of texture is among the most salient haptic characteristics of objects. It can be a compelling cue to object identity and it can strongly influence forces during manipulation [KL02]. In medical applications with limited visual feedback, such as minimally invasive or endoscopic surgery [Sal99], and virtual prototyping applications of mechanical assembly and maintainability assessment [WM03], accurate haptic feedback of surface detail is a key factor for successful dexterous operations.

To be correctly represented, surfaces with high-frequency geometric texture detail require higher sampling densities, thereby increasing the cost of collision detection. In fact, computation of texture-induced forces using full-resolution geometric representations of the objects and handling contacts at microgeometric scale is computationally prohibitive, and novel representations must be considered. Similar to graphical texture rendering [Cat74], researchers in haptic rendering have investigated geometric representations where objects with high combinatorial complexity (i.e., with a high polygon count) are described by coarse representations along with texture images that store fine geometric detail.

This chapter begins with a summary of texture rendering methods for 3-DoF haptic rendering. Then we introduce a force model for collision response between two textured surfaces, which captures perceptually relevant aspects identified in psychophysics studies. We conclude the chapter with the description of a fast algorithm for GPU-based computation of approximate penetration depth, which enables 6-DoF haptic texture rendering in combination with the force model for textured surfaces.

4.1 THREE-DOF HAPTIC TEXTURE RENDERING

Three-DoF haptic rendering algorithms have been extended to account for subfeature geometric detail that is not directly encoded in the geometric primitives, in a way similar to the texture mapping technique broadly employed in computer graphics. Indeed, haptic rendering of textures was one of the first tackled problems in the field of computational haptics by Minsky et al. [MOyS+90]. This section begins with a description of Minsky's pioneering algorithm

for rendering textures on the plane [Min95]. Then it discusses rendering of textures on 3D surfaces, covering offset-based methods and probabilistic methods.

4.1.1 Rendering Textures on the Plane

Minsky [Min95] developed the *Sandpaper* system for 2-DoF haptic rendering of textures on a planar surface. Her system was built around a force model for computing 2D forces from texture height field information. Following energy-based arguments, her force model synthesizes a force $\mathbf{F}$ in 2D based on the gradient of the texture height field h at the location of the probe:

$$\mathbf{F} = -k\nabla h. \tag{4.1}$$

Minsky also qualitatively and quantitatively analyzed roughness perception and the believability of the proposed force model. One of the main conclusions of her work was to establish her initial hypothesis that texture information can be conveyed by displaying forces tangential to the contact surface. This hypothesis was later exploited for rendering textured 3D surfaces [HBS99].

4.1.2 Methods Based on Surface Offsets

High-resolution surface geometry can be represented by a parameterized coarse mesh along with texture images storing detailed offset or displacement field information, similarly to the common approach of texture mapping in computer graphics [Cat74]. Constraint-based 3-DoF haptic rendering methods determine a unique contact point on the surface of the rendered object. Usually, the mesh representation used for determining the contact point is rather coarse and does not capture high-frequency texture. Nevertheless, the parametric coordinates of the contact point can be used for accessing surface texture information from texture images.

Ho et al. [HBS99] introduced a technique similar to bump mapping [Bli78] that alters the surface normal based on the gradient of the texture offset field. A combination of the original and refined normals is used for computing the direction of the feedback force.

Techniques for haptic texture rendering based on a single contact point can capture geometric properties of only one object and are not suitable for simulating full interaction between two surfaces. The geometric interaction between two surfaces is not limited to, and cannot be described by, a pair of contact points. Moreover, the local kinematics of the contact between two surfaces include rotational degrees of freedom, which are not captured by point-based methods.

Ho et al. [HBS99] indicated that a high offset gradient can induce system instability. Along a similar direction, Choi and Tan [CT03b, CT03a] studied the influence of collision detection and penetration depth computation on 3-DoF haptic texture rendering. Discontinuities

in the output of collision detection are perceived by the user a phenomenon that they described as *aliveness*. This phenomenon is a possible problem in 6-DoF haptic rendering as well.

4.1.3 Probabilistic Methods

Some researchers have exploited statistical properties of surfaces for computing texture-induced forces that are added to the classic 3-DoF contact forces. Siira and Pai [SP96] synthesized texture forces according to a Gaussian distribution for generating a sensation of roughness. In order to improve stability, they did not apply texture forces during static contact. Later, Pai et al. [PvdDJ+01] presented a technique for rendering roughness effects by dynamically modifying the coefficient of friction of a surface. The roughness-related portion of the friction coefficient was computed according to an autoregressive process driven by noise.

Probabilistic methods have proved to be successful for rendering high-frequency roughness effects in point-surface contact. It is also possible, although this approach has yet to be explored, that they could be combined with geometric techniques for synthesizing high-frequency effects in 6-DoF haptic rendering.

4.2 PERCEPTUALLY DRIVEN FORCE MODEL

In this section we describe a force model for 6-DoF haptic texture rendering. First we describe some design considerations. Then, we detail the force and torque equations based on the gradient of directional penetration depth and discuss the solution of the gradient using finite differences.

4.2.1 Offset Surfaces and Penetration Depth

As summarized in Section 1.3.2, Klatzky and Lederman [KL02] concluded, after a series of studies, that the perception of roughness is intimately related to the trajectory of the probe grabbed by the user. For spherical probes as those used in their studies, and in the absence of dynamic effects, the surface traced during exploration constitutes an offset surface. The oscillation of the offset surface produces the vibratory motion that encodes roughness. The idea of offset surfaces has also been used by Okamura and Cutkosky [OC01] to model interaction between robotic fingers and textured surfaces.

The height h of the offset surface traced by a spherical probe is equal to the vertical penetration depth δ if the center of the sphere moves exactly along the surface. This connection between penetration depth and offset surfaces can be generalized to nonspherical probes through the concept of Minkowski sum. An offset surface corresponds to the boundary of the Minkowski sum of a given surface and a sphere. Therefore, the height of the offset surface at a particular point is the distance to the boundary of the Minkowski sum for a particular position of the probe, which is the same as the penetration depth. Actually, the height of the offset surface is the distance to the surface along a particular direction (i.e., vertical), so the distance to the

boundary of the Minkowski sum must also be measured along a particular direction. This distance is known to be the *directional penetration depth*.

Since, for spherical probes, perception of roughness is tightly coupled with the undulation of the traced offset surface, a texture force model for general surfaces should take into account the variation of penetration depth (i.e., its gradient). As noted earlier, the gradient of a height field has also been used in the context of 3-DoF rendering methods [Min95, HBS99] as a descriptor for texture-induced forces. The use of the gradient of penetration depth in 6-DoF haptic rendering can be considered as a generalization of the concept used in 3-DoF haptic rendering.

4.2.2 Penalty-Based Texture Force

Otaduy and Lin [OL04] designed a force model for collision response between textured surfaces that would account for the effects of geometry and normal force identified in Klatzky and Lederman's perceptual studies. As haptic rendering is a human-in-the-loop system, dynamic effects associated with grasping factors, such as exploratory speed, need not be modeled explicitly. The force model extends classic penalty-based collision response (see Section 2.1.2) by defining an elastic penetration energy U with stiffness k:

$$U = \frac{1}{2} k \delta^2. \tag{4.2}$$

Based on this energy, texture force $\mathbf{F}$ and torque $\mathbf{T}$ are defined as

$$\begin{pmatrix} \mathbf{F} \\ \mathbf{T} \end{pmatrix} = -\nabla U = -k\delta \left(\nabla \delta \right), \tag{4.3}$$

where $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}, \frac{\partial}{\partial \theta_x}, \frac{\partial}{\partial \theta_y}, \frac{\partial}{\partial \theta_z} \right)$ is the gradient in 6-DoF configuration space.

Each contact between two objects A and B can be described by a pair of contact points $\mathbf{p}_A$ and $\mathbf{p}_B$, and by a penetration direction $\mathbf{n}$. The penetration depth between objects A and B can be locally approximated by the directional penetration depth $\delta_\mathbf{n}$ along $\mathbf{n}$. Then, Eq. (4.3) is rewritten for $\delta_\mathbf{n}$ in a reference system $\{\mathbf{u}, \mathbf{v}, \mathbf{n}\}$ located at the center of mass of A. The axes $\mathbf{u}$ and $\mathbf{v}$ may be selected arbitrarily as long as they form an orthonormal basis with $\mathbf{n}$. Equation. (4.3) reduces to

$$\left(F_u \; F_v \; F_n \; T_u \; T_v \; T_n \right)^T = -k\delta_\mathbf{n} \left(\frac{\partial \delta_\mathbf{n}}{\partial u} \; \frac{\partial \delta_\mathbf{n}}{\partial v} \; 1 \; \frac{\partial \delta_\mathbf{n}}{\partial \theta_u} \; \frac{\partial \delta_\mathbf{n}}{\partial \theta_v} \; \frac{\partial \delta_\mathbf{n}}{\partial \theta_n} \right)^T, \tag{4.4}$$

where θ_u, θ_v, and θ_n are the rotation angles around the axes $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{n}$ respectively.

The force and torque on object A (and similarly on object B) for each contact can be expressed in the global reference system as

$$\begin{aligned}\mathbf{F}_A &= (\mathbf{u}\ \mathbf{v}\ \mathbf{n})\,(F_u\ F_v\ F_n)^T,\\ \mathbf{T}_A &= (\mathbf{u}\ \mathbf{v}\ \mathbf{n})\,(T_u\ T_v\ T_n)^T.\end{aligned} \tag{4.5}$$

Forces and torques of all contacts are summed up to compute the net force and torque.

Generalizing Minsky's approach for 3-DoF haptic rendering [Min95], the tangential forces F_u and F_v are proportional to the gradient of penetration depth. However, the 6-DoF force model also defines a penalty-based normal force and gradient-dependent torque that describe full 3D object–object interaction. In addition, the tangential force and the torque are proportional to the normal force, which is consistent with the results of psychophysics studies, showing that perceived roughness increases with the magnitude of the normal force [KL02].

4.2.3 Gradient of Penetration Depth

Penetration depth functions δ and $\delta_{\mathbf{n}}$ are sampled at discrete points on a 6-DoF configuration space. With central differencing, the partial derivatives can be approximated as

$$\frac{\partial \delta_{\mathbf{n}}}{\partial u} = \frac{\delta_{\mathbf{n}}(u+\Delta u, v, n, \theta_u, \theta_v, \theta_n) - \delta_{\mathbf{n}}(u-\Delta u, v, n, \theta_u, \theta_v, \theta_n)}{2\Delta u}, \tag{4.6}$$

and similarly for $\frac{\partial \delta_{\mathbf{n}}}{\partial v}$, $\frac{\partial \delta_{\mathbf{n}}}{\partial \theta_u}$, $\frac{\partial \delta_{\mathbf{n}}}{\partial \theta_v}$, and $\frac{\partial \delta_{\mathbf{n}}}{\partial \theta_n}$.

$\delta_{\mathbf{n}}(u+\Delta u, \ldots)$ can be obtained by translating object A a distance Δu along the $\mathbf{u}$-axis and computing the directional penetration depth. A similar procedure is followed for other penetration depth values.

4.3 PENETRATION DEPTH BETWEEN TEXTURED MODELS

Otaduy et al. [OJSL04] designed a 6-DoF haptic texture rendering algorithm in which geometric models are composed of simplified representations along with texture images storing fine geometric detail. In the context of haptic rendering, these texture images are referred to as *haptic textures*. Fig. 4.1 depicts an example with a hammer and a CAD part, and the haptic texture for the hammer.

The main idea behind the haptic texture rendering approach is a two-stage algorithm for computing penetration depth, which is then used to apply collision response with the texture force model described earlier. This two-stage algorithm is described in detail later, but it can be summarized as follows:

1. Obtain approximate contact information from simplified geometric representations.
   a) Perform collision detection between the low-resolution meshes.
   b) Identify each pair of intersecting surface patches as *one contact*.

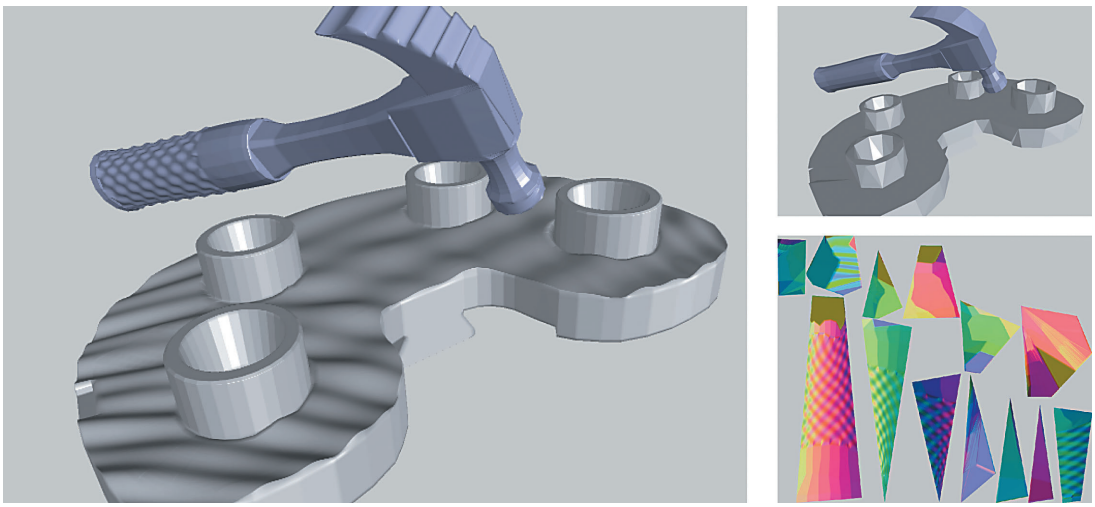

FIGURE 4.1: *Haptic rendering of interaction between textured models.* Top: high-resolution textured hammer (433k polygons) and CAD part (658k polygons). Bottom left: low-resolution models (518 and 720 polygons). Bottom right: hammer texture with fine geometric detail [OJSL04] (©2004 IEEE).

c) Characterize each contact by a pair of contact points on the patches and a penetration direction $\mathbf{n}$.

2. Refine this contact information using detailed geometric information stored in haptic textures.
   a) For each contact, compute approximate directional penetration depth along $\mathbf{n}$, using haptic textures.
   b) Compute force and torque, using the force model for texture rendering described in the previous section.

4.3.1 Definitions of Directional Penetration Depth

As described in Section 3.4, the penetration depth δ between two intersecting polyhedra A and B is typically defined as the minimum translational distance required for separating them, and this distance is equivalent to the distance from the origin to the Minkowski sum of A and $-B$. On the other hand, the *directional penetration depth* $\delta_{\mathbf{n}}$ along the direction $\mathbf{n}$ is defined as the minimum translation along $\mathbf{n}$ to separate the polyhedra.

The algorithm for computing the directional penetration depth assumes that the intersecting surface patches can be represented as height fields along the penetration direction. A *height field* H is defined as a set $H = \{(x, y, z) \in \mathbb{R}^3 \mid z = h(x, y)\}$. $h : \mathbb{R}^2 \to \mathbb{R}$ is called a *height function*. Let $\mathbf{p}$ denote a point in $\mathbb{R}^3$, let $\mathbf{p}_{xyz} = (p_x \; p_y \; p_z)^T$ denote the coordinates of

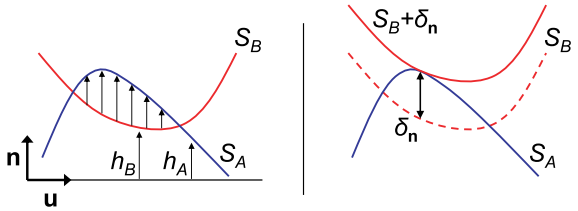

FIGURE 4.2: *Penetration depth of height fields.* Directional penetration depth of surface patches expressed as height difference [OJSL04] (©2004 IEEE).

$\mathbf{p}$ in a global reference system, and $\mathbf{p}_{uvn} = (p_u \;\; p_v \;\; p_n)^T$ its coordinates in a rotated reference system $\{\mathbf{u}, \mathbf{v}, \mathbf{n}\}$. A surface patch $S \subset \mathbb{R}^3$ can be represented as a height field along a direction $\mathbf{n}$, if $p_n = h(p_u, p_v), \forall \mathbf{p} \in S$. Then, one can define a mapping $g : D \to S$, $D \subset \mathbb{R}^2$, as $g(p_u, p_v) = \mathbf{p}_{xyz}$, where

$$\mathbf{p}_{xyz} = g(p_u, p_v) = (\mathbf{u} \;\; \mathbf{v} \;\; \mathbf{n})\,(p_u \;\; p_v \;\; h(p_u, p_v))^T. \tag{4.7}$$

The inverse of the mapping g is the orthographic projection of S onto the plane $(\mathbf{u}, \mathbf{v})$ along direction $\mathbf{n}$. Given the mapping g, the height function h can be computed as

$$h(p_u, p_v) = \mathbf{n} \cdot g(p_u, p_v). \tag{4.8}$$

For two intersecting surface patches S_A and S_B that can be represented as height fields along a direction $\mathbf{n}$, their directional penetration depth $\delta_\mathbf{n}$ is the maximum height difference along direction $\mathbf{n}$, as illustrated in Fig. 4.2 by a 2D example.

A parameterization of the surface patches by orthographic projection along $\mathbf{n}$ yields mappings $g_A : D_A \to S_A$ and $g_B : D_B \to S_B$, as well as height functions $h_A : D_A \to \mathbb{R}$ and $h_B : D_B \to \mathbb{R}$. Therefore, the directional penetration depth $\delta_\mathbf{n}$ can be defined as

$$\delta_\mathbf{n} = \max_{(u,v)\in(D_A \cap D_B)} (h_A(u, v) - h_B(u, v)). \tag{4.9}$$

4.3.2 Two-Stage Algorithm

Each contact between objects A and B is defined by two intersecting surface patches S_A and S_B. Using a geometric representation that combines low-resolution meshes and haptic textures, the surface patch S_A is approximated by a low-resolution surface patch $\hat{S}_A$ (and similarly for S_B). The function $f_A : \hat{S}_A \to S_A$ defines a mapping from the low-resolution surface patch $\hat{S}_A$ to the surface patch S_A.

Collision detection between the two low-resolution surfaces patches $\hat{S}_A$ and $\hat{S}_B$ returns a penetration direction $\mathbf{n}$. Given a rotated reference system $\{\mathbf{u}, \mathbf{v}, \mathbf{n}\}$ and assuming that all S_A, $\hat{S}_A$, S_B, and $\hat{S}_B$ can be represented as height fields along $\mathbf{n}$, S_A and $\hat{S}_A$ are projected

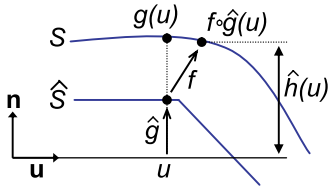

FIGURE 4.3: *Approximate height function.* Height function of a surface patch approximated by a composite mapping function [OJSL04] (© 2004 IEEE).

orthographically along **n** onto the plane (**u**, **v**). This projection yields mappings $g_A : D_A \to S_A$ and $\hat{g}_A : \hat{D}_A \to \hat{S}_A$. One can define $\bar{D}_A = D_A \cap \hat{D}_A$.

The mapping function g_A can be approximated by a composite mapping function $f_A \circ \hat{g}_A : \bar{D}_A \to S_A$ (see Fig. 4.3). From Eq. (4.8), an approximate height function $\hat{h}_A : \bar{D}_A \to \mathbb{R}$ is defined as

$$\hat{h}_A(u, v) = \mathbf{n} \cdot (f_A \circ \hat{g}_A(u, v)). \tag{4.10}$$

Given approximate height functions $\hat{h}_A$ and $\hat{h}_B$, a domain $D = \bar{D}_A \cap \bar{D}_B$, and Eq. (4.9), the directional penetration depth $\delta_{\mathbf{n}}$ of S_A and S_B can be approximated by

$$\hat{\delta_{\mathbf{n}}} = \max_{(u,v)\in D} \left(\hat{h}_A(u, v) - \hat{h}_B(u, v)\right). \tag{4.11}$$

Even though the computation of $\hat{\delta_{\mathbf{n}}}$ can be realized on CPUs, this algorithm is best suited for implementation on graphics processors (GPUs), as discussed next.

4.3.3 Computation on Graphics Hardware

As shown in Eq. (4.4), computation of 3D texture-induced force and torque according to the texture force model requires the computation of directional penetration depth $\delta_{\mathbf{n}}$ and its gradient at every contact. From Eq. (4.6), this requirement reduces to computing $\delta_{\mathbf{n}}$ all together at 11 configurations of object A. Due to the use of central differencing to compute partial derivatives of $\delta_{\mathbf{n}}$, object A must be transformed to two different configurations, where $\delta_{\mathbf{n}}$ is recomputed. All together the force model requires the computation of $\delta_{\mathbf{n}}$ itself and five partial derivatives, hence 11 configurations. As pointed out in Section 3.4, computation of penetration depth using exact object-space or configuration-space algorithms is too expensive for haptic rendering applications. Instead, the approximation $\hat{\delta_{\mathbf{n}}}$ according to Eqs. (4.10) and (4.11) leads to a natural and efficient image-based implementation on programmable graphics hardware. The mappings $\hat{g}$ and f correspond, respectively, to orthographic projection and texture mapping operations, which are most suited for parallel and grid-based computation using GPUs.

For every contact, first one must compute $\hat{h}_B$ and then perform two operations for each of the 11 object configurations: (1) compute $\hat{h}_A$ for the transformed object A, and (2) find the penetration depth $\hat{\delta}_{\mathbf{n}} = \max(\Delta\hat{h}) = \max\left(\hat{h}_A - \hat{h}_B\right)$. The height difference at the actual object configuration is denoted by $\Delta\hat{h}(0)$, and the height differences at the transformed configurations by $\Delta\hat{h}(\pm\Delta u)$, $\Delta\hat{h}(\pm\Delta v)$, $\Delta\hat{h}(\pm\Delta\theta_u)$, $\Delta\hat{h}(\pm\Delta\theta_v)$, and $\Delta\hat{h}(\pm\Delta\theta_n)$.

4.3.3.1 Computation of Height Functions

In the GPU-based implementation, the mapping $f : \hat{S} \rightarrow S$ is implemented as a texture map (i.e., haptic texture) that stores geometric detail of the high-resolution surface patch S. The mapping $\hat{g}$ is implemented by rendering $\hat{S}$ using an orthographic projection along $\mathbf{n}$. The height function $\hat{h}$ is computed in a fragment program. Points in S are obtained by looking up the haptic texture f and projecting the position onto $\mathbf{n}$. The result is stored in a floating point texture t.

Geometric texture mapping is chosen over other methods for approximating h (e.g., rendering S directly or performing displacement mapping) in order to maximize performance. The input haptic texture f is stored as a floating point texture.

4.3.3.2 Search of Maximum Values

The max function in Eq. (4.11) could be implemented as a combination of frame buffer read-back and CPU-based search. Expensive read-backs, however, can be avoided by posing the max function as a binary search on the GPU [GLW+04]. Given two height functions $\hat{h}_A$ and $\hat{h}_B$ stored in textures t_1 and t_2, their difference is computed and stored in the depth buffer. Then the height difference is scaled and offset to fit in the depth range. Height subtraction and copy to depth buffer are performed in a fragment program, by rendering a quad that covers the entire buffer. For a depth buffer with N bits of precision, the search domain is the integer interval $[0, 2^N)$. The binary search starts by querying if there is any value larger than 2^{N-1}. A quad is rendered at depth 2^{N-1} and an occlusion query is performed (see http://www.nvidia.com/dev_content/nvopenglspecs/GL_NV_occlusion_query.txt), which will report if any pixel passed the depth test, i.e., the stored depth was larger than 2^{N-1}. Based on the result, the depth of a new quad is set, and the binary search continues.

4.3.3.3 Gradient Computation

The height functions $\hat{h}_A(\pm\Delta u)$, $\hat{h}_A(\pm\Delta v)$, and $\hat{h}_A(\pm\Delta\theta_n)$ may be obtained by simply translating or rotating $\hat{h}_A(0)$. As a result, only six height functions $\hat{h}_A(0)$, $\hat{h}_B(0)$, $\hat{h}_A(\pm\Delta\theta_u)$, and $\hat{h}_A(\pm\Delta\theta_v)$ need to be computed for each pair of contact patches. These six height functions are tiled in one single texture t to minimize context switches and increase performance (see Fig. 4.4).

Moreover, the domain of each height function is split into four quarters, each of which is mapped to one of the RGBA channels. This optimization exploits vector computation

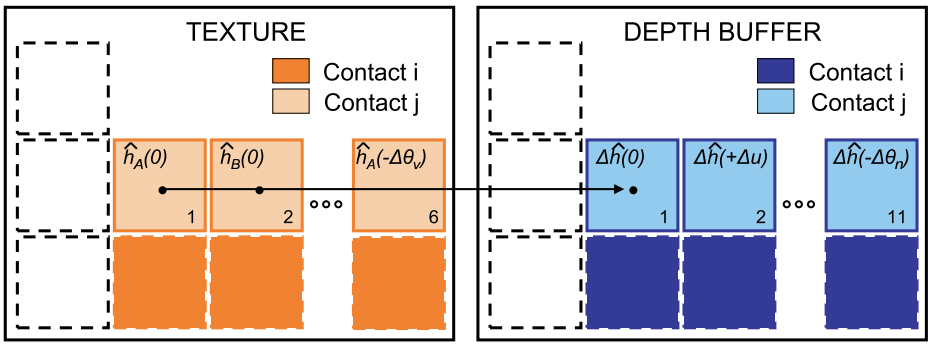

FIGURE 4.4: *Tiling in the GPU.* Tiling of multiple height functions and contacts to minimize context switches between target buffers [OJSL04] (©2004 IEEE).

capabilities of fragment processors. As shown in Fig. 4.4, one can also tile 11 height differences per contact in the depth buffer.

4.3.3.4 Multiple Simultaneous Contacts

The computational cost of haptic texture rendering increases linearly with the number of contacts between the interacting objects. However, performance can be further optimized. In order to limit context switches, the height functions associated with multiple pairs of contact patches are tiled in one single texture t and the height differences are tiled in the depth buffer as well, as shown in Fig. 4.4. The cost of *max search* operations is further minimized by performing occlusion queries on all contacts in parallel.

4.4 DISCUSSION

Otaduy and Lin [OL04] demonstrated through a series of experiments with a simulated model that there is a qualitative match between the effects produced by the force model described in Section 4.2 and the results of the studies on roughness perception directed by Klatzky and Lederman [KL02]. Specifically, the effects of probe diameter and applied force on the acceleration of a simulated hand induced by the force model match the effects of these factors on perceived roughness of real textures in a qualitative way. The results exhibit some differences on the effects of exploratory speed, but these differences may be caused by limitations of the dynamic hand model employed in the experiments. The complete connection between physical parameters, such as forces and motion, and a subjective metric of roughness is still unknown. Nevertheless, the analysis of simulated accelerations and perceived roughness reflects high correlation of the locations and values of function maxima.

Despite the apparent validity of the texture force model and the high performance achieved with the GPU-based computation of penetration depth, 6-DoF haptic texture rendering still

presents some limitations, and should be a topic for further research. An important issue in every force model for haptic rendering is its stability. Choi and Tan [CT03a] have shown that even passive rendering algorithms may suffer from a problem called *aliveness*, induced by geometric discontinuities. Using haptic textures, discontinuities may arise if the contact patches cannot be described as height fields along the penetration direction, and these are possible sources of aliveness.

Also, as with other discrete techniques, the haptic texture rendering algorithm is susceptible to aliasing problems. Some of the potential aliasing sources are low resolution of the input textures, low spatial resolution in the image-based computation of penetration depth, approximation of derivatives with central differencing, and temporal sampling.

In some contact scenarios with large contact areas, the definition of a local and directional penetration depth is not applicable. An example is the problem of screw insertion. In situations with contact between interlocking features, local geometry cannot be represented as height fields and the gradient of directional penetration depth may not capture the interlocking effects.

In practice, the force model generates forces that create a realistic perception of roughness for object–object interaction; however, one essential limitation of penalty-based collision response is the inability to enforce motion constraints. The texture force model attempts to do so by increasing tangential contact stiffness when the gradient of penetration depth is high. But the stiffness delivered to the user must be limited, for stability purposes. New constraint-based haptic rendering techniques and perhaps other haptic devices [PC99] will be required to properly enforce constraints.

CHAPTER 5

Future Directions

It has been more than 40 years since Ivan Sutherland suggested an *ultimate display* with force feedback [Sut65], and we have steadily moved toward this goal in the past decades. Many researchers in computer graphics have investigated the problem of interactive graphical rendering for increasingly complex objects and scenes over the years. Today virtual environments present synthetic images of objects with complex shapes, highly textured surfaces, and rich lighting effects. More and more researchers are also becoming aware of the importance of *touch* in virtual enviornments and have been working on various issues for haptic display as well. Although in the 1990s haptic rendering techniques targeted mostly at the problem of tracing objects with a point (i.e., 3-DoF haptic rendering), recent advances have laid the algorithmic foundation for interactive 6-DoF haptic display of fairly complex surfaces. This synthesis has presented the state-of-the-art techniques that focus on overcoming the high computational cost of contact determination between complex models, while satisfying the real-time constraints of force feedback. As discussed earlier, 6-DoF haptic rendering methodologies rely on a combination of efficient algorithms on collision detection and rigid-body simulation, and basic knowledge of control theory. Moreover, by developing advanced multiresolution haptic rendering algorithms based on the psychophysics of touch, it is possible to achieve high-fidelity 6-DoF haptic rendering of complex models consisting of hundreds of thousands of geometric primitives.

However, there remain several research challenges for 6-DoF haptic rendering. This chapter briefly discusses some open problems in the design of force feedback devices, rendering of deformable models, issues related to the evaluation and validation of rendering methodologies, and possible novel applications of 6-DoF haptic rendering.

5.1 FORCE FEEDBACK DEVICES

Most of the earlier force feedback hardware has been rather heavy and can be awkward to wear. In addition, due to intricate mechanical design and low production quantity, it also tends to be rather expensive, which in turn limits its use in consumer products and its applicability in

practical situations where haptics can make a significant impact. An excellent survey by Burdea of the different classes of force feedback devices can be found in [Bur96].

However, the more recent devices, starting from PHANToM [MS94], have improved their ergonomics noticeably. With improved software support and promising application development, haptic devices are increasingly used in several different areas. Among the existing commercial products, one of the most rudimentary form and commonly used force feedback mechanism is vibratory feedback of various degrees of sophistication and often available on commodity products, ranging from mobile phones, haptic mice for desktop systems, to video game controllers. Other more recent examples where force feedback mechanisms have already impacted the design of human–system interfaces include active control in the automobile industry and touch screens with vibrotactile feedback.

For 6-DoF haptic rendering, devices are currently quite limited in terms of workspace and delivered force range due to various tradeoffs in mechanical design. Prices are high, thus they still have a rather small market presence. But, there is a growing number of commercially available devices [HGA+98, Che99, And02, Rup00], which together with a recent reduction of prices on some 3-DoF devices, and significant improvements on the rendering methodologies and application software, make haptic rendering a more viable and cost-effective way for enabling "virtual touch" in various forms and places. We expect that the use of 6-DoF haptic devices will also steadily become prevailing in the coming years.

Perhaps the next wave of device design will focus more on glove-based, light-weight, wearable tactile feedback interfaces [Bur96]. Such devices have tremendous potential in virtual palpation procedures, feeling various surface properties (textures, temperature, friction, etc.) and providing a wider range of support for assisted technology.

5.2 DEFORMABLE BODIES

Many real-world objects undergo deformations and topological changes, and a complete set of 6-DoF haptic rendering algorithms should account for these physical phenomena as well. However, most of the existing algorithms for haptic rendering make assumptions about the rigidity of objects, both for the acceleration of collision detection and for the simplification of dynamics simulation.

Current collision detection techniques based on bounding volume hierarchies, though very effective for rigid bodies, are not necessarily well suited for deformable bodies, due to the cost of updating the precomputed data structures. Bounding volumes with fast update operations (e.g., spheres or AABBs) are perhaps worth considering, but other more complex bounding volumes (e.g., OBBs or convex hulls), which lead to superlinear cost for updating the hierarchies, do not seem to be applicable. Some possible directions for addressing the problem of collision detection between deformable bodies include hierarchical data structures

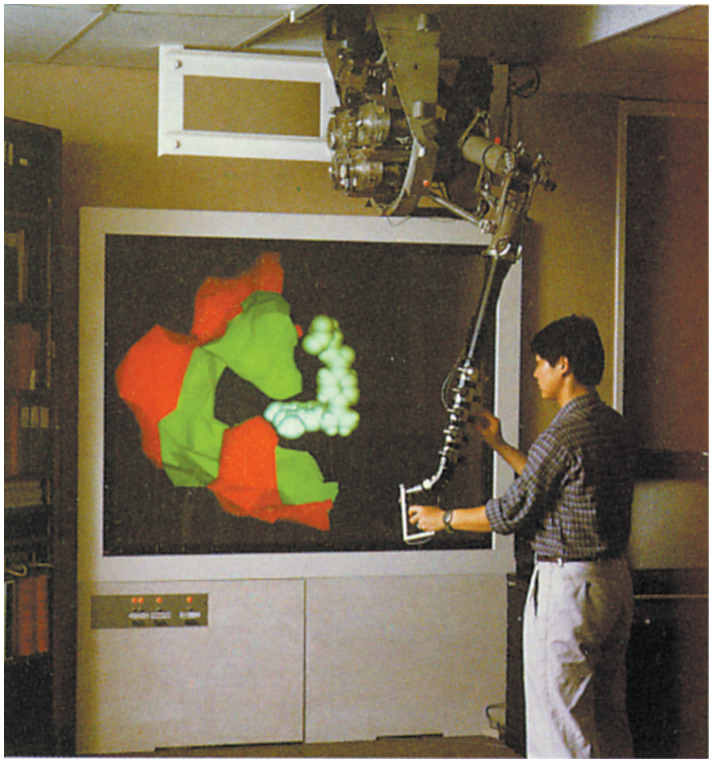

FIGURE 5.1: *Molecular docking.* Image of 6-DoF haptic rendering system in use, courtesy of Brooks et al. [BOYBK90], University of North Carolina at Chapel Hill (©ACM, 1990).

without bottom-up updates such as BD-trees [JP04], or more general techniques based on spatial partitioning [THM+03], visibility tests on GPUs [GRLM03], and distance fields (especially those that can be computed at interactive rates) [FL01, HZLM01, FPRJ00, SOM04]. Probably the most challenging aspect lies in fast and robust handling of self-collisions. Although some solutions exist [TKH+05, GKJ+05], they cannot meet the hard performance requirements of 6-DoF haptic rendering.

Haptic rendering of the interaction between deformable bodies poses essential difficulties due to the large number of degrees of freedom that must be simulated. This topic has been covered extensively in computational mechanics and also in computer graphics (see surveys [GM97, NMK+05]), but the complexity of the models that can be simulated currently at interactive rates for graphical display is still rather limited. Some techniques have been proposed to perform 6-DoF haptic rendering of moderately complex deformable objects [JP99, DAK04a, DAK04b, BJ05], but they typically make restrictive assumptions on the nature of the deformations (e.g., for quasirigid behavior or few dominant DoFs), to reduce

the cost of collision detection and contact handling. Handling topological changes with force feedback is especially important in surgical simulation, and some researchers have developed force models that capture tissue failure [Gre98]. However, computational models for general, physically based fracture [OH99, MH01, OBH02] are not yet efficient enough for haptic rendering.

Exploiting observations drawn from psychophysics research has proved to be extremely beneficial for the design of efficient 6-DoF haptic rendering of rigid bodies. Perhaps, in the design of efficient rendering algorithms for deformable bodies, we can continue to derive computational principles and benefit from the known facts on human tactile perception as well.

5.3 EVALUATION AND VALIDATION

The effectiveness of the different haptic rendering methodologies, both existing or those to be developed, should be analyzed from the perspective of human factors. For instance, it would be interesting to analyze stability and transparency of the rendering methodologies, the influence on task performance of the different components of the complete rendering pipeline, the effectiveness in conveying properties such as stiffness, roughness, or friction, and the effects of visual and haptic discrepancies.

Further investigation on human kinesthetic perception will undoubtedly be beneficial for overall research in haptic rendering. As discussed earlier, the development of efficient (possibly adaptive) techniques for computing the interaction with dynamic and/or deformable bodies requires studies on human perception that will guide the design of error metrics and force models.

5.4 POTENTIAL APPLICATIONS

A natural way of assessing the effectiveness of the 6-DoF haptic rendering methodologies presented in this synthesis will be to incorporate them into practical applications.

5.4.1 Scientific Visualization

Perhaps one of the earliest and most convincing force feedback applications is the display of force fields among molecules for docking a drug molecule with its receptor site in a protein [BOYBK90]. This system, shown in Fig. 5.1, was developed at the University of North Carolina, Chapel Hill, and it computed forces and torques between molecules arising from electrostatic and other interatomic forces, and then displayed them to the users to identify minimum-energy configurations. The experimental results and reports from the scientists who used the system indicate promising benefits of 6-DoF force feedback.

Another successful example is the development of the nanomanipulator application (see Fig. 5.2) that provides an intuitive VR interface to scanning-probe microscopes with force

FIGURE 5.2: *UNC nanomanipulator system.* A virtual reality interface for manipulating nanostructures [TRC+93, SFL+06] (©2006 IEEE).

feedback control [TRC+93]. The incorporation of force display enables the scientists to feel the nanosurfaces and conduct new experiments otherwise not possible.

Techniques for haptic visualization of the topology of vector fields have also been investigated earlier by Helman and Hesselink [HH90, HH91]. Durbeck et al. [DMW+98] have also described a system for enhancing scientific visualization by the use of haptic feedback. The

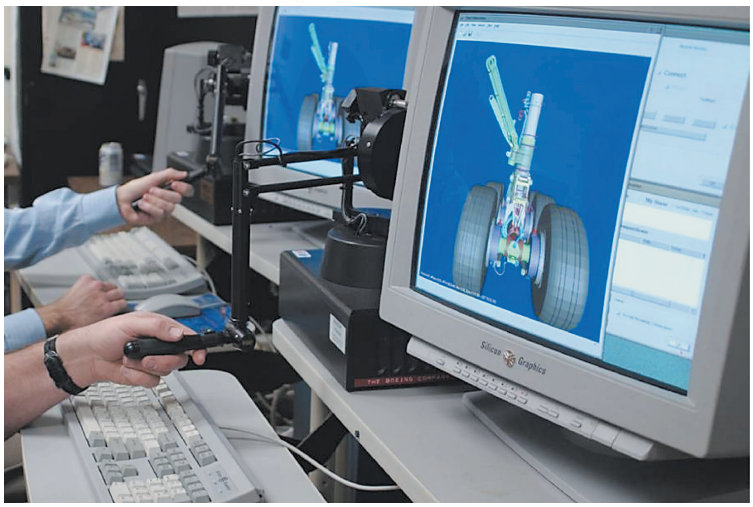

FIGURE 5.3: *Assembly planning in aircraft design.* Use of 6-DoF haptic rendering with the VPS system. Image courtesy of McNeely et al. [MPT06] Boeing (Printed with permission of Haptics-e).

FIGURE 5.4: *Haptic painting system setup.* An artist using a haptic stylus to paint directly on the virtual canvas using dAb [BSLM01, LBF+02] (©2002 IEEE).

combined haptics/graphics display is used for displaying flow fields and vector fields. These systems were based on 3-DOF haptic devices that provided force feedback only. Lawrence et al. presented a technique for shock and vortex visualization using a combined visual and haptic interface with a 5-DOF force feedback device [LLPN00]. Most recently, Baxter and Lin [BL04] presented a general haptic rendering method for interacting with fluid media. These projects are just a few examples of early exploration of haptics for scientific visualization. Much remains to be investigated for other applications of haptic rendering in scientific exploration.

5.4.2 Engineering Design and Prototyping

The main benefit of virtual reality lies in the fact that it provides high bandwidth, multisensory, and real-time interaction between a user and the computer. Haptic rendering provides direct perception of three-dimensional (3D) objects and directly couples input and output between the computer and the user. It allows designers to reach out and feel their conceptual designs, manipulate and, modify the geometric models interactively. The synergy of force feedback and graphical display enables designers to interact more intuitively and naturally with virtual prototyping environments or human-augmented systems for effective design verification and a rapid manufacturing process. Such augmented systems have been used for some engineering design and prototyping applications; for example, manipulation of virtual mechanisms [NNHJ98], object grasping [MH98], evaluating car dashboard designs [SBC97], experiencing assembly forces [RGW97], virtual prototyping [HCT+97]), and assembly planning in the aircraft design [MPT99, MPT06]. A wider adaptation of haptic technology may stimulate more research

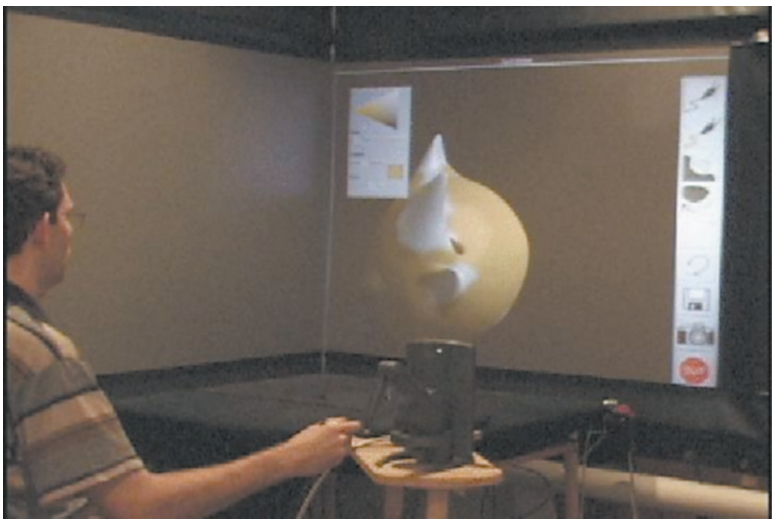

FIGURE 5.5: *Haptic modeling and painting system setup.* A user creating a model on a large display with ArtNova [FOL02, LBF+02] (©2002 IEEE).

in the 6-DoF force display of interaction between sculptured models and result in shortened product development cycles and reduced prototyping costs.

5.4.3 Medical Training

Another application, and perhaps a driving force of research in haptic rendering these days, is surgical simulation. Most of the surgical procedures where training simulators are being conceived involve interaction with soft tissue. Existing methods have been mostly limited to haptic rendering with few degrees of freedom and more advanced rendering techniques will be required for truly 6-DoF manipulation of soft tissues. However, sinus surgery is a notable exception where a training simulator clearly calls for 6-DoF haptic rendering of both hard structures and soft tissues. In addition, during sinus surgical procedures, surgeons receive visual feedback from endoscopic cameras that are not aligned with their view direction. The images provided by the cameras are difficult to interpret by surgeons, and they often conflict with the haptic cues provided by the interaction of the tools and the sinus cavities. This application will also be an excellent case study on coordination and conflicts among multisensory cues.

The development of haptic rendering techniques for applications such as sinus surgery simulation also introduces new challenges. Virtual tools cannot always be considered as a simple rigid body, but perhaps an articulated body, which implies the addition of constraints to the computational models. Most importantly, the interaction paradigm of virtual coupling, which considers grasping the virtual tool at one single point, may no longer be valid, and new interaction paradigms may be necessary.

5.4.4 3D Model Design

Another exciting application of force feedback is 3D model design, including modeling, sculpting, and painting. Most of the existing commercial computer systems for modeling, sculpting, painting, and drawing use just the 2D input and output devices typical of current desktop computing environments. They often lack the capability of direct 3D interaction. Even if users can directly manipulate the image on screen, their movements at any one time are limited by the number of degrees of freedom of the input device. The resulting high learning curves sometimes deter the users from freely expressing their creativity due to the difficulty in translating conceptual designs into digital forms. Furthermore, existing commercial computer systems and recent research on the automatic generation of digital forms have mainly emphasized the appearance of the final products and not the process itself.

Some recent commercial and research systems have introduced the capability of force feedback into 3D modeling, scuplting, and painting systems. For example, FreeForm©, and ClayTools—virtual scultping and modeling systems developed by SensAble Technologies, inTouch [GEL00], and ArtNova [FOL02] (see Fig. 5.5)—touch-enabled 3D painting and multiresolution modeling systems, and dAb (see Fig. 5.4)—haptic painting with 3D deformable brushes [BSLM01].

Based on the user feedback, the addition of haptic interaction can considerably improve the ease and expressiveness of 3D model design systems and enhance user's experiences in digitial design and creative applications. They also introduce a new set of research challenges, including real-time physically based modeling of various elements in the system, better integration of haptic and visual cues, as well as incorporation of nonphysical attributes (such as anticipation of a new event).

Bibliography

[AGHP+00] P. Agarwal, L. Guibas, S. Har-Peled, A. Rabinovitch, and M. Sharir, "Penetration depth of two convex polytopes in 3d", *Nordic J. Comput.*, Vol. 7, pp. 227–240, 2000.

[AH98a] R. J. Adams and B. Hannaford, "A two-port framework for the design of unconditionally stable haptic interfaces", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1998.

[AH98b] O. R. Astley and V. Hayward, "Multirate haptic simulation achieved by coupling finite element meshes through norton equivalents", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 1998.

[AKO95] Y. Adachi, T. Kumano, and K. Ogino, "Intermediate representation for stiff virtual objects", *Virtual Reality Annual Int. Symp.*, 1995, pp. 203–210.

[And02] C. Andriot, Advances in virtual prototyping. *Clefs CEA, Vol 47, Research and Simulation*, 2002.

[AS96] R. S. Avila and L. M. Sobierajski, "A haptic interaction method for volume visualization", in *IEEE Visualization '96*. IEEE, Oct. 1996.

[AWD+04] B. Adams, M. Wicke, P. Dutre, M. Gross, M. Pauly, and M. Teschner, "Interactive 3d painting on point-sampled objects", *Eurographics Symp. on Point-Based Graphics*, 2004.

[Bar89] D. Baraff, "Analytical methods for dynamic simulation of non-penetrating rigid bodies", in *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, Jeffrey Lane, ed., July 1989, pp. 223–232.

[Bar91] D. Baraff, "Coping with friction for non-penetrating rigid body simulation", in *Computer Graphics (SIGGRAPH '91 Proc.)*, Vol. 25, T. W. Sederberg, ed., July 1991, pp. 31–40.

[Bar92] D. Baraff, "Dynamic simulation of non-penetrating rigid body simulation", Ph.D. thesis, Cornell University, Ithica, New York, 1992.

[Bar94] D. Baraff, "Fast contact force computation for nonpenetrating rigid bodies", in *Proc. of SIGGRAPH '94*, Andrew Glassner, ed., ACM SIGGRAPH, 1994, pp. 23–34.

[Ber99] P. J. Berkelman, "Tool-Based Haptic Interaction with Dynamic Physical Simulations Using Lorentz Magnetic Levitation", Ph.D. thesis, Carnegie Mellon University, Pittsburg, PA, 1999.

[BHS97] C. Basdogan, C.-H. Ho, and M. A. Srinivasan, "A ray-based haptic rendering technique for displaying shape and texture of 3d objects in virtual environments", *DSC-Vol. 61, Proc. of the ASME Dynamic Systems and Control Division*, 1997, pp. 77–84.

[BJ05] J. Barbič and D. L. James, "Real-time subspace integration of St. Venant-Kirchhoff deformable models", in *Proc. of ACM SIGGRAPH*, 2005.

[BKSS90] N. Beckmann, H. Kriegel, R. Schneider, and B. Seeger, "The r*-tree: an efficient and robust access method for points and rectangles", *Proc. SIGMOD Conf. on Management of Data*, 1990, pp. 322–331.

[BL04] W. Baxter and M. Lin, "Haptic interaction with fluid media", in *Proc. of Graphics Interface '04*, May 2004.

[Bli78] J. F. Blinn, "Simulation of wrinkled surfaces", in *Computer Graphics (SIGGRAPH '78 Proceedings)*, 1978, pp. 286–292.

[BOYBK90] F. P. Brooks, Jr., M. Ouh-Young, J. J. Batter, and P. J. Kilpatrick, "Project GROPE—Haptic displays for scientific visualization", in Forest Baskett, ed., *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, August 1990, pp. 177–185.

[BS80] A. Bejczy and J. K. Salisbury, "Kinematic coupling between operator and remote manipulator", *Adv. Comput. Technol.*, Vol. 1, 1980, pp. 197–211.

[BSLM01] W. Baxter, V. Scheib, M. Lin, and D. Manocha, "DAB: Haptic painting with 3D virtual brushes", in *Proc. of ACM SIGGRAPH*, 2001, pp. 461–468.

[Bur96] G. Burdea, *Force and Touch Feedback for Virtual Reality*. New Wiley, 1996.

[BW01] D. Baraff and A. Witkin, *Physically-Based Modeling*. (ACM SIGGRAPH Course Notes). 2001.

[Cam97] S. Cameron, "Enhancing GJK: Computing minimum and penetration distance between convex polyhedra", *IEEE Int. Conf. on Robotics and Automation*, 1997, pp. 3112–3117.

[Cat74] E. E. Catmull. "A Subdivision Algorithm for Computer Display of Curved Surfaces", Ph.D. thesis, Dept. of CS, University of Utah, Salt lake city, UT, 1974.

[CB94] J. E. Colgate and J. M. Brown, "Factors affecting the z-width of a haptic display", *IEEE Int. Conf. on Robotics and Automation*, 1994, pp. 3205–3210.

[CC86] S. Cameron and R. K. Culley, "Determining the minimum translational distance between two convex polyhedra", in *Proc. of Int. Conf. on Robotics and Automation*, 1986, pp. 591–596.

[CC97] B. Chang and J. E. Colgate, "Real-time impulse-based simulation of rigid body systems for haptic display", in *Proc. of ASME Dynamic Systems and Control Division*, 1997.

[CD68] R. W. Cottle and G. B. Dantzig, "Complementarity pivot theory of mathematical programming", *Linear Algebr. Appl.*, 1968, Vol. 1, pp. 103–125.doi:org/10.1016/0024-3795(68)90052-9

[CDST97] B. Chazelle, D. Dobkin, N. Shouraboura, and A. Tal, "Strategies for polyhedral surface decomposition: An experimental study", *Comput. Geom.: Theory Appl.*, 1997, Vol. 7, pp. 327–342.

[CGSS93] J. E. Colgate, P. E. Grafing, M. C. Stanley, and G. Schenkel, "Implementation of stiff virtual walls in force-reflecting interfaces", in *Virtual Reality Annual Int. Symp.*, 1993, pp. 202–207.

[Che99] E. Chen, "Six degree-of-freedom haptic system for desktop virtual prototypin applications", in *Proc. of the First Int. Workshop on Virtual Reality and Prototyping*, 1999, pp. 97–106.

[CJ92] C. E. Connor and K. O. Johnson, "Neural coding of tactile texture: Comparison of spatial and temporal mechanisms for roughness perception", *J. Neurosci.*, Vol. 12, pp. 3414–3426, 1992.

[CLMP95] J. D. Cohen, M. C. Lin, D. Manocha, and M. K. Ponamgi, "I-collide: An interactive and exact collision detection system for large-scale environments", *in Proc. ACM Interactive 3D Graphics Conf.*, 1995, pp. 189–196.

[CpS92] R. W. Cottle, J. S. Pang, and R. E. Stone, *The linear complementarity problem*. New York: Academic-Press, 1992.

[CS94] J. E. Colgate and G. G. Schenkel, "Passivity of a class of sampled-data systems: Application to haptic interfaces", in *Proc. of American Control Conf.*, 1994.

[CSB95] J. E. Colgate, M. C. Stanley, and J. M. Brown, "Issues in the haptic display of tool use", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1995, pp. 140–145.

[CSC04] D. Constantinescu, S. E. Salcudean, and E. A. Croft, "Impulsive forces for haptic rendering of rigid contact", in *Proc. of Int. Symp. on Robotics*, 2004, pp. 1–6.

[CSC05] D. Constantinescu, S. E. Salcudean, and E. A. Croft, "Haptic rendering of rigid contacts using impulsive and penalty forces", *IEEE Transactions on Robotics*, Vol. 21, No. 3, pp. 309–323, 2005.doi:org/10.1109/TRO.2004.840906

[cT00] M. C. Çavuşoğlu and F. Tendick, "Multirate simulation for high fidelity haptic interaction with deformable objects in virtual environments", *in Proc. of IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 2458–2465.

[CT03a] S. Choi and H. Z. Tan, "Aliveness: Perceived instability from a passive haptic texture rendering system", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003.

[CT03b] S. Choi and H. Z. Tan, "An experimental study of perceived instability during haptic texture rendering: Effects of collision detection algorithm", in *Proc. of Haptics Symp.*, 2003, pp. 197–204.

[cTS02] M. C. Çavuşoğlu, F. Tendick, and S. S. Sastry, "Haptic interfaces to real and virtual surgical environments", in *Touch in Virtual Environments*, M. L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme, eds., Chapter 13, pp. 217–237. Upper Saddle River, NJ: Prentice Hall, 2002.

[DAK04a] C. Duriez, C. Andriot, and A. Kheddar, "A multi-threaded approach for deformable/rigid contacts with haptic feedback.", in *Proc. of Haptics Symp.*, 2004.

[DAK04b] C. Duriez, C. Andriot, and A. Kheddar, "Signorini's contact model for deformable objects in haptic simulations", in *Proc. of IEEE/RSJ IROS*, 2004.

[DHKS93] D. Dobkin, J. Hershberger, D. Kirkpatrick, and S. Suri, "Computing the intersection-depth of polyhedra", *Algorithmica*, Vol. 9, pp. 518–533, 1993.doi:org/10.1007/BF01190153

[DK90] D. P. Dobkin and D. G. Kirkpatrick, "Determining the separation of preprocessed polyhedra—a unified approach", in *Proc. 17th Int. Colloquin Automata Lang. Program.*, *Lecture Notes in Computer Science* 1990, Vol. 443, pp. 400–413. Berlin: Springer.

[DMW+98] L. Durbeck, N. Macias, D. Weinstein, C. Johnson, and J. Hollerbach, "Scirun haptic display for scientific visualization", in *Phantom Users Group Meetings*, 1998.

[DQ+99] F. Dachille, H. Qin, , A. Kaufman, and J. El-Sana, "Haptic sculpting of dynamic surfaces", in *Proc. of ACM Symp. on Interactive 3D Graphics*, 1999, pp. 103–110.

[Dru97] S. Druyan, "Effect of the kinesthetic conflict on promoting scientific reasoning", *J. Res. Sci. Teach.*, Vol. 34, pp. 1083–1099, 1997.doi:org/10.1002/(SICI)1098-2736(199712)34:10<1083::AID-TEA7>3.0.CO;2-N

[EB01] M. O. Ernst and M. S. Banks, "Does vision always dominate haptics?", in *Touch in Virtual Environments Conf.*, 2001.

[EHS+97] C. Edmond, D. Heskamp, D. Sluis, D. Stredney, G. Wiet, R. Yagel, S. Weghorst, P. Oppenheimer, J. Miller, M. Levin, and L. Rosenberg, "Ent endoscopic surgical simulator", in *Proc. of Medicine Meets VR*, 1997, pp. 518–528.

[EL00] S. Ehmann and M. C. Lin, "Accelerated proximity queries between convex polyhedra using multi-level voronoi marching", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2000, pp. 2101–2106.

[EL01] S. Ehmann and M. C. Lin, "Accurate and fast proximity queries between polyhedra using convex surface decomposition", *Comput. Graph. Forum*, Vol. 20, No. 3, pp. 500–510, 2001.doi:org/10.1111/1467-8659.00543

[Erl04] K. Erleben, "Stable, Robust, and Versatile Multibody Dynamics Animation", Ph.D. thesis, University of Copenhagen, Copenhagen, Denmark, 2004.

[ESJ97] R. E. Ellis, N. Sarkar, and M. A. Jenkins, "Numerical methods for the force reflection of contact", *ASME Trans. Dyn. Syst., Model. Control*, Vol. 119, pp. 768–774, 1997.

[ESV00] J. El-Sana and A. Varshney, "Continuously-adaptive haptic rendering", in *Virtual Environments 2000*, 2000, pp. 135–144.

[FFM+04] B. Fisher, S. Fels, K. MacLean, T. Munzner, and R. Rensink, "Seeing, hearing and touching: Putting it all together", in *ACM SIGGRAPH course notes*, 2004.

[FL01] S. Fisher and M. C. Lin, "Fast penetration depth estimation for elastic bodies using deformed distance fields", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems 2001*, 2001.

[FOL02] M. Foskey, M. A. Otaduy, and M. C. Lin, "ArtNova: touch-enabled 3D model design", in *Proc. of IEEE Virtual Reality Conf.*, 2002.

[FPRJ00] S. Frisken, R. Perry, A. Rockwood, and R. Jones, "Adaptively sampled distance fields: A general representation of shapes for computer graphics", in *Proc. of ACM SIGGRAPH*, 2000, pp. 249–254.

[GBF03] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking", *ACM Trans. Graph.*, Vol. 22, pp. 871–878, 2003.doi:org/10.1145/882262.882358

[GEL00] A. Gregory, S. Ehmann, and M. C. Lin, "*inTouch*: interactive multiresolution modeling and 3d painting with a haptic interface", in *Proc. of IEEE VR Conf.*, 2000.

[GH97] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics", in *Proc. of ACM SIGGRAPH*, 1997, pp. 209–216.

[GHL05] M. Glencross, R. Hubbold, and B. Lyons, "Dynamic primitive caching for haptic rendering of large-scale models", in *Proc. of World Haptic Conference* 2005, pp. 517–518.

[GHZ99] L. Guibas, D. Hsu, and L. Zhang, "*H-Walk*: Hierarchical distance computation for moving convex bodies", in *Proc. of ACM Symp. on Computational Geometry*, 1999.

[GJK88] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi. "A fast procedure for computing the distance between objects in three-dimensional space", *IEEE J. Robot. Autom.*, Vol. RA-4, PP. 193–203, 1988.

[GKJ+05] N. K. Govindaraju, D. Knott, N. Jain, I. Kabul, R. Tamstoff, R. Gayle, M. C. Lin, and D. Manocha, "Interactive collision detection between deformable models using chromatic decomposition", in *Proc. of ACM SIGGRAPH*, 2005.

[GLGT99] A. Gregory, M. Lin, S. Gottschalk, and R. Taylor, "H-COLLIDE: A framework for fast and accurate collision detection for haptic interaction", in *Proc. of Virtual Reality Conf. 1999*, 1999, pp. 38–45.

[GLM96] S. Gottschalk, M. Lin, and D. Manocha. OBB-Tree: A hierarchical structure for rapid interference detection. *Proc. of ACM Siggraph'96*, 1996, pp. 171–180.doi:org/full_text

[GLW+04] N. Govindaraju, B. Lloyd, W. Wang, M. Lin, and D. Manocha, "Fast computation of database operations using graphics processors", in *Proc. of ACM SIGMOD*, 2004.

[GM97] S. F. Gibson and B. Mirtich, "A survey of deformable modeling in computer graphics", Technical Report, Mitsubishi Electric Research Laboratory, 1997.

[GME+00] A. Gregory, A. Mascarenhas, S. Ehmann, M. C. Lin, and D. Manocha. "6-DoF haptic display of polygonal models", in *Proc. of IEEE Visualization Conf.*, 2000.

[Got00] S. Gottschalk, "Collision Queries using Oriented Bounding Boxes", Ph.D. thesis, Department of Computer Science, University of North Carolina, Chapel, Hill, NC, 2000.

[Gre98] S. Greenish, "Acquisition and analysis of cutting forces of surgical instruments for haptic simulation", M.Sc. thesis, McGill University, Quebic, Canada, 1998.

[GRLM03] N. Govindaraju, S. Redon, M. Lin, and D. Manocha, "CULLIDE: Interactive collision detection between complex models in large environments using graphics hardware", in *Proc. of ACM SIGGRAPH/Eurographics Workshop on Graphics Hardware*, 2003, pp. 25–32.

[GSM+97] S. Gibson, J. Samosky, A. Mor, C. Fyock, E. Grimson, and T. Kanade, "Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback", *First Joint Conf. on Computer Vision, Virtual Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery (CVMed-MRCAS)*, 1997, pp. 368–378.

[GSS99] I. Guskov, W. Sweldens, and P. Schroder, "Multiresolution signal processing for meshes", in *Proc. of ACM SIGGRAPH*, 1999, pp. 325–334.

[GT54] R. Goertz and R. Thompson, "Electronically controlled manipulator", *Nucleonics*, Vol. 12, No. 11, pp. 46–47, 1954.

[HA00] V. Hayward and B Armstrong. "A new computational model of friction applied to haptic rendering", in *Experimental Robotics VI*, P. P. Corke and J. Trevelyan Eds., New York: Springer, pp. 404-412, 2000.

[HBS99] C.-H. Ho, C. Basdogan, and M. A. Srinivasan. "Efficient point-based rendering techniques for haptic display of virtual objects", *Presence*, Vol. 8, No. 5, pp. 477–491, 1999.

[HCGB99] M. A. Heller, J. A. Calcaterra, S. L. Green, and L. Brown. "Intersensory conflict between vision and touch: The response modality dominates when precise, attention-riveting judgements are required", *Perception and Psychophysics*, Vol. 61, pp. 1384–1398, 1999.

[HCT+97] J. Hollerbach, E. Cohen, W. Thompson, R. Freier, D. Johnson, A. Nahvi, D. Nelson, and T. Thompson II, "Haptic interfacing for virtual prototyping of mechanical CAD designs", *CDROM Proc. of ASME Design for Manufacturing Symp.*, 1997.

[HGA+98] V. Hayward, P. Gregorio, O. Astley, S. Greenish, and M. Doyon. "Freedom-7: A high fidelity seven axis haptic device with applications to surgical training", in *Experimental Robotics*. London: Springer 1998, pp. 445–456.

[HH90] J. Helman and L. Hesselink. "Representation and display of vector field topology in fluid flow data sets", in *Visualization in scientific computing*, M. Neilson and B.D. Shimer, eds., Los Alamitos CA: IEEE Computers Society 1990, pp. 61–73.

[HH91] J. L. Helman and L. Hesslink. Surface representations of two- and three-dimensional Fluid flow topology. in *Visualization '91*, October 1991, pp. 6–13.

[Hog85] N. Hogan. "Impedance control: An approach to manipulation, part i—theory, part ii—implementation, part iii—applications", *J. Dyn. Syst., Meas. Control*, Vol. 107, pp. 1–24, 1985.

[Hog86] N. Hogan, "Multivariable mechanics of the neuromuscular system", *IEEE Annual Conf. of the Engineering in Medicine and Biology Society*, 1986, pp. 594–598.

[Hop96] H. Hoppe. "Progressive meshes", in *SIGGRAPH 96 Conf. Proc., (Annual Conf. Series)* Holly Rushmeier, ed. pp. 99–108. Readings, MA: Addison Wesley, August 1996.

[Hop97] H. Hoppe, "View dependent refinement of progressive meshes", *ACM SIGGRAPH Conf. Proc.*, 1997, pp. 189–198.

[HR00] M. Hollins and S. R. Risner. "Evidence for the duplex theory of tactile texture perception", *Perception & Psychophysics*, Vol. 62, pp. 695–705, 2000.

[HRK02] B. Hannaford, J.-H. Ryu, and Y. S. Kim, "Stable control of haptics", in *Touch in Virtual Environments*, M. L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme, eds. Chapter 3, pp. 47–70. Upper Saddle River, NJ: Prentice Hall, 2002.

[HS77] J. W. Hill and J. K. Salisbury, "Two measures of performance in a peg-in-hole manipulation task with force feedback", *Thirteenth Annual Conf. on Manual Control, MIT*, 1977.

[Hub94] P. Hubbard. *Collision Detection for Interactive Graphics Applications*. Ph.D. thesis, Brown University, Providence, R1, 1994.

[HZLM01] K. Hoff, A. Zaferakis, M. Lin, and D. Manocha, "Fast and simple 2d geometric proximity queries using graphics hardware", in *Proc. of ACM Symp. on Interactive 3D Graphics*, 2001, pp. 145–148.

[IMWB01] B. Insko, M. Meehan, M. Whitton, and F. Brooks, "Passive haptics significantly enhances virtual environments", Technical Report 01-010, Department of Computer Science, University of North Carolina, Chapel Hill NC, 2001.

[Ins01] B. Insko. *Passive Haptics Significantly Enhance Virtual Environments*. Ph.D. thesis. Department of Computer Science, University of North Carolina. Chapel Hill, NC, 2001.

[JC01] D. E. Johnson and E. Cohen. Spatialized normal cone hierarchies. in *Proc. of ACM Symp. on Interactive 3D Graphics*, pp. 129–134, 2001.doi:org/full_text

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. "Data clustering: a review", *ACM Comput. Surv.*, Vol. 31, No. 3, pp. 264–323, 1999.doi:org/10.1145/331499.331504

[JP99] D. L. James and D. K. Pai, "Artdefo: accurate real time deformable objects", in *Proc. of ACM SIGGRAPH*, 1999.

[JP04] D. L. James and D. K. Pai. "Bd-tree: output-sensitive collision detection for reduced deformable models", *ACM Trans. on Graph.*, Vol. 23, No. 3, 2004.

[JTK+99] D. Johnson, T. V. Thompson II, M. Kaplan, D. Nelson, and E. Cohen, "Painting textures with a haptic interface", in *Proc. of IEEE Virtual Reality Conf.*, 1999.

[JW03] D. E. Johnson and P. Willemsen, "Six degree of freedom haptic rendering of complex polygonal models", in *Proc. of Haptics Symp.*, 2003.

[JW04] D. E. Johnson and P. Willemsen, "Accelerated haptic rendering of polygonal models through local descent", in *Proc. of Haptics Symp.*, 2004.

[JWC05] D. E. Johnson, P. Willemsen, and E. Cohen. "6-dof haptic rendering using spatialized normal cone search", *IEEE Trans. Visual. Comput. Graph.*, Vol. 11, No. 6, pp. 661–670, 2005.doi:org/10.1109/TVCG.2005.106

[Kar85] D. Karnopp. "Computer simulation of stick slip friction in mechanical dynamic systems", *Trans. ASME, J. Dyn. Syst., Meas. Control*, Vol. 107, pp. 100–103, 1985.

[Kat89] D. Katz. *The World of Touch*. Erlbaum, Hillsdale, NJ, 1989. L. Krueger, Trans. (Original work published 1925).

[KB91] W. Kim and A. Bejczy, "Graphical displays for operator aid in telemanipulation", *IEEE Int. Conf. on Systems, Man and Cybernetics*, 1991.

[KEP05] Danny M. Kaufman, Timothy Edmunds, and Dinesh K. Pai, "Fast frictional dynamics for rigid bodies", in *Proc. of ACM SIGGRAPH*, 2005.

[KHM+98] J. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan. "Efficient collision detection using bounding volume hierarchies of k-dops", *IEEE Trans. Visual. Comput. Graph.*, Vol. 4, No. 1, pp. 21–37, 1998.

[Kil76] P. J. Kilpatrick. *The use of a kinesthetic supplement in an interactive graphics system*. Ph.D. thesis, The University of North Carolina Chapel Hill NC, 1976.

[KKH+97] U. G. Kuhnapfel, C. Kuhn, M. Hubner, H.-G. Krumm, H. Maass, and B. Neisius. "The karlsruhe endoscopic surgery trainer as an example for virtual reality in medical education", *Minimally Invasive Ther. Allied Technol.*, Vol. 6, pp. 122–125, 1997.

[KL95] R. L. Klatzky and S. J. Lederman. "Identifying objects from a haptic glance", *Perception and Psychophysics*, Vol. 57, pp. 1111–1123, 1995.

[KL99] R. L. Klatzky and S. J. Lederman. Tactile roughness perception with a rigid link interposed between skin and surface. *Perception and Psychophysics*, Vol. 61, pp. 591–607, 1999.

[KL02] R. L. Klatzky and S. J. Lederman. "Perceiving texture through a probe", in M. L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme, eds, *Touch in Virtual Environments*, chapter 10, pp. 180–193. Upper Saddle River, NJ: Prentice Hall, 2002.

[KL03] R. L. Klatzky and S. J. Lederman. "Touch", in *Experimental Psychology*, pp. 147–176, 2003. Vol. 4 in I.B. Weiner. Ed. Handbook of Psychology. New York: Wiley

[KLH+03] R. L. Klatzky, S. J. Lederman, C. Hamilton, M. Grindley, and R. H. Swendsen. "Feeling textures through a probe: Effects of probe and surface geometry and exploratory factors", *Perception and Psychophysics*, Vol. 65, No. 4, pp. 613–631, 2003.

[KLM02a] Y. J. Kim, M. C. Lin, and D. Manocha, "DEEP: an incremental algorithm for penetration depth computation between convex polytopes", in *Proc. of IEEE Conf. on Robotics and Automation*, 2002, pp. 921–926.

[KLM02b] Y. J. Kim, M. C. Lin, and D. Manocha, "Fast penetration depth computation using rasterization hardware and hierarchical refinement", in *Proc. of Workshop on Algorithmic Foundations of Robotics*, 2002.

[KLM04] Y. J. Kim, M. Lin, and D. Manocha. "Incremental penetration depth estimation between convex polytopes using dual-space expansion", *IEEE Trans. on Visual. Comput. Graph.*, Vol. 10, No. 1, pp. 152–164, 2004.

[KOLM03] Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha. "Six-degree-of-freedom haptic rendering using incremental and localized computations", *Presence*, Vol. 12, No. 3, pp. 277–295, 2003.doi:org/10.1162/105474603765879530

[L84] P. Lötstedt. Numerical simulation of time-dependent contact friction problems in rigid body mechanics. *SIAM J. of Sci. Stat. Comput.*, 5, pp. 370–393, 1984.

[Lar01] E. Larsen, "A robot soccer simulator: a case study for rigid body contact", *Game Developers Conf.*, 2001.

[LBF+02] M. C. Lin, W. Baxter, M. Foskey, M. A. Otaduy, and V. Scheib, "Haptic interaction for creative processes with simulated media", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2002.

[LC91] M. C. Lin and J. F. Canny, "Efficient algorithms for incremental distance computation", in *Proc. IEEE Internat. Conf. Robot. Autom.*, 1991, Vol. 2, pp. 1008–1014.

[LCN99] J. C. Lombardo, M.-P. Cani, and F. Neyret, "Real-time collision detection for virtual surgery", in *Proc. of Computer Animation*, 1999.

[LE97] D. Luebke and C. Erikson, "View-dependent simplification of arbitrary polygon environments", in *Proc. of ACM SIGGRAPH*, 1997.

[Led74] S. J. Lederman. "Tactile roughness of grooved surfaces: The touching process and the effects of macro- and microsurface structure", *Perception and Psychophysics*, Vol. 16, pp. 385–395, 1974.

[Lem65] C. E. Lemke. "Bimatrix equilibrium points and mathematical programming", *Management Science*, Vol. 11, pp. 681–689, 1965.

[LG98] M. Lin and S. Gottschalk, "Collision detection between geometric models: a survey", in *Proc. of IMA Conf. on Mathematics of Surfaces*, 1998.

[LGLM00] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha, "Distance queries with rectangular swept sphere volumes", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2000.

[Lin93] M. C. Lin. *Efficient Collision Detection for Animation and Robotics*. Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, 1993.

[LKHG00] S. J. Lederman, R. L. Klatzky, C. Hamilton, and M. Grindley, "Perceiving surface roughness through a probe: Effects of applied force and probe diameter", in *Proc. of the ASME DSCD-IMECE*, 2000.

[LKHR99] S. J. Lederman, R. L. Klatzky, C. Hamilton, and G. I. Ramsay. "Perceiving roughness via a rigid stylus: Psychophysical effects of exploration speed and mode of touch", *Haptics-e Electron. J. Haptic Res.*, Vol. 1, No. 1, 1999.

[Llo57] S. P. Lloyd. "Least squares quantization in PCM'S", *Bell Telephone Labs Memo*, 1957.

[LLPN00] D. A. Lawrence, C. D. Lee, L. Y. Pao, and R. Y. Novoselov, "Shock and vortex visualization using a combined visual/haptic interface", in *Proc. of IEEE Visualization*, 2000, pp. 131–137.

[LM04] M. C. Lin and D. Manocha. Collision and proximity queries. In J. E. Goodman and J. O'Rourke, eds., *Handbook of Discrete and Computational Geometry, 2nd Ed.*, chapter 35, pp. 787–807. CRC Press, Boca Raton, FL, 2004.

[LS91] R. H. LaMotte and M. A. Srinivasan, "Surface microgeometry: tactile perception and neural encoding", In O. Franzen and J. Westman, eds., *Information Processing in the Somatosensory System*, pp. 49–58. Macmillan, London, 1991.

[LX04] Q. Luo and J. Xiao. "Physically accurate haptic rendering with dynamic effects", *IEEE Comput. Graph. Appl.*, Vol. 24, No. 6, pp. 60–69, 2004.doi:org/10.1109/MCG.2004.59

[Mau03] S. Mauch. *Efficient Algorithms for Solving Static Hamilton-Jacobi Equations*. Ph.D. thesis, Californa Institute of Technology, Pasadena, CA, 2003.

[MC95] B. Mirtich and J. Canny, "Impulse-based simulation of rigid bodies", *Symp. on Interactive 3D Graphics*. ACM Press, 1995.

[MCF99] B. E. Miller, J. E. Colgate, and R. A. Freeman. "Guaranteed stability of haptic systems with nonlinear virtual environments", *IEEE Trans. Robot. Autom.*, Vol. 16, No. 6, pp. 712–719, 1999.

[MGC96] N. Miner, R. B. Gillespie, and T. Caudell, "Examining the influence of audio and visual stimuli on a haptic display", in *Proc. of IMAGE Conf.*, 1996.

[MH98] H. Maekawa and J. Hollerbach, "Haptic display for object grasping and manipulation in virtual environment", *IEEE Conf. on Robotics and Automation*, 1998, pp. 2566–25725733.

[MH01] M. Mahvash and V. Hayward. Haptic rendering of cutting: A fracture mechanics approach. *haptics-e*, Vol. 2, No. 3, 2001.

[MH05] M. Mahvash and V. Hayward. "High-fidelity passive force-reflecting virtual environments", *IEEE Trans. Robot.*, Vol. 21, No. 1, pp. 38–46, 2005.doi:org/10.1109/TRO.2004.833819

[MHS02] M.L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme. *Touch in Virtual Environments*. Prentice Hall, 2002.

[Min95] M. Minsky. *Computational Haptics: The Sandpaper System for Synthesizing Texture for a Force-Feedback Display*. Ph.D. thesis, Ph.D. Dissertation, Program in Media Arts and Sciences, MIT, Cambridge, MA 1995. Thesis work done at UNC-CH Computer Science.

[Mir96] B. V. Mirtich. *Impulse-based Dynamic Simulation of Rigid Body Systems*. Ph.D. thesis, University of California, Berkeley, CA, 1996.

[Mir98] B. Mirtich, "V-Clip: Fast and robust polyhedral collision detection", *ACM Trans. Graph.*, Vol. 17, No. 3, pp. 177–208, July 1998.doi:org/10.1145/285857.285860

[Mir00] B. Mirtich, "Timewarp rigid body simulation", *SIGGRAPH 00 Conf. Proc.*, 2000, pp. 193–200.

[MOyS+90] M. Minsky, M. Ouh-Young, O. Steele, F. P. Brooks, Jr., and M. Behensky. Feeling and seeing: Issues in force display. In Rich Riesenfeld and Carlo Sequin, eds., *Computer Graphics (1990 Symp. on Interactive 3D Graphics)*, Vol. 24, pp. 235–243, March 1990.

[MPT99] W. McNeely, K. Puterbaugh, and J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling", in *Proc. of ACM SIGGRAPH*, 1999, pp. 401–408.

[MPT06] W. McNeely, K. Puterbaugh, and J. Troy. "Voxel-based 6-dof haptic rendering improvements", *Haptics-e*, Vol. 3, No. 7, 2006.

[MQW01] K. McDonnell, H. Qin, and R. Wlodarczyk, " Virtual clay: a real-time sculpting system with haptic interface", in *Proc. of ACM Symp. on Interactive 3D Graphics*, 2001, pp. 179–190.

[MRF+96] W. Mark, S. Randolph, M. Finch, J. van Verth, and R. M. Taylor II, "Adding force feedback to graphics systems: Issues and solutions", in *SIGGRAPH 96 Conf. Proc., (Annual Conf. Series),* Holly Rushmeier, ed. pp. 447–452, 1996.

[MS94] T. M. Massie and J. K. Salisbury, "The phantom haptic interface: a device for probing virtual objects", in *Proc. of ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 1994, 1:295–301.

[MS01] V. J. Milenkovic and H. Schmidl, "Optimization-based animation", *SIGGRAPH01 Conf. Proc.*, 2001, pp. 37–46.

[MW88] M. Moore and J. Wilhelms. "Collision detection and response for computer animation", in John Dill, ed., *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, pp. 289–298, August 1988.

[NJC99] D. D. Nelson, D. E. Johnson, and E. Cohen, "Haptic rendering of surface-to-surface sculpted model interaction", in *Proc. of ASME Dynamic Systems and Control Division*, 1999.

[NMK+05] A. Nealen, M. Müller, R. Keiser, E. Boxermann, and M. Carlson, "Physically based deformable models in computer graphics (state of the art report)", *Eurographics STAR*, 2005.

[NNHJ98] A. Nahvi, D. Nelson, J. Hollerbach, and D. Johnson, "Haptic manipulation of virtual mechanisms from mechanical CAD designs", in *Proc. of IEEE Conf. on Robotics and Automation*, 1998, pp. 375–380.

[OBH02] J. F. O'Brien, A. W. Bargteil, and J. K. Hodgins, "Graphical modeling and animation of ductile fracture", in *Proc. of ACM SIGGRAPH*, 2002, pp. 291–294.

[OC99] A. Okamura and M. Cutkosky, "Haptic exploration of fine surface features", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 1999, pp. 2930–2936.

[OC01] A. M. Okamura and M. R. Cutkosky. "Feature detection for haptic exploration with robotic fingers", *Int. J. Robot. Res.*, Vol. 20, No. 12, pp. 925–938, 2001.doi:org/10.1177/02783640122068191

[OD01] C. O'Sullivan and J. Dingliana. "Collisions and perception", *ACM Trans. Graph.*, Vol. 20, No. 3, pp. 151–168, 2001.doi:org/10.1145/501786.501788

[ODGK03] C. O'Sullivan, J. Dingliana, T. Giang, and M. K. Kaiser, "Evaluating the visual fidelity of physically based animations", in *Proc. of ACM SIGGRAPH*, 2003, pp. 527–536.

[OH99] J. F. O'Brien and J. K. Hodgins, "Graphical modeling and animation of brittle fracture", in *Proc. of ACM SIGGRAPH*, 1999.

[OJSL04] M. A. Otaduy, N. Jain, A. Sud, and M. C. Lin, "Haptic display of interaction between textured models", in *Proc. of IEEE Visualization*, 2004, pp. 297–304.

[OL03a] M. A. Otaduy and M. C. Lin, "CLODs: Dual hierarchies for multiresolution collision detection", *Eurographics Symp. on Geometry Processing*, 2003, pp. 94–101.doi:org/10.1145/882262.882305

[OL03b] M. A. Otaduy and M. C. Lin, "Sensation preserving simplification for haptic rendering", *ACM Trans. on Graph.*, Vol. 22, pp. 543–553, 2003.

[OL04] M. A. Otaduy and M. C. Lin, "A perceptually-inspired force model for haptic texture rendering", in *Proc. of Symp. APGV*, 2004, pp. 123–126.

[OL05] M. A. Otaduy and M. C. Lin, "Stable and responsive six-degree-of-freedom haptic manipulation using implicit integration", in *Proc. of World Haptics Conf.*, 2005.

[ORC99] C. O'Sullivan, R. Radach, and S. Collins. "A model of collision perception for real-time animation" *Computer Animation and Simulation*, pp. 67–76, 1999.

[ORC06] M. Ortega, S. Redon, and S. Coquillart, "A six degree-of-freedom god-object method for haptic display of rigid bodies", in *Proc. of IEEE Virtual Reality Conf.*, 2006.

[OY90] M. Ouh-Young, "Force Display in Molecular Docking", Ph.D. thesis, Computer Science Department, University of North Carolina, Chapel Hill, NC, 1990.

[PC99] M. Peshkin and J. E. Colgate. "Cobots", *Industrial Robot*, Vol. 26, No. 5, pp. 335–341, 1999.

[PJC04] K. Potter, D. Johnson, and E. Cohen, "Height field haptics", in *Proc. of Haptics Symp.*, 2004.

[PR97] D. K. Pai and L. M. Reissel. "Haptic interaction with multiresolution image curves", *Comput. Graph.*, Vol. 21, pp. 405–411, 1997.doi:org/10.1016/S0097-8493(97)00031-9

[PvdDJ+01] D. K. Pai, K. van den Doel, D. L. James, J. Lang, J. E. Lloyd, J. L. Richmond, and S. H. Yau. Scanning physical interaction behavior of 3d objects. *Computer Graphics (Proc. of ACM SIGGRAPH)*, 2001.

[Qui94] S. Quinlan, "Efficient distance computation between non-convex objects", in *Proc. of Int. Conf. on Robotics and Automation*, 1994, pp. 3324–3329.

[RGW97] T. Sheridan R. Gupta and D. Whitney, "Experiments using multimodal virtual environments", *Presence*, Vol. 6, pp. 318–338, 1997.

[RK00] D. Ruspini and O. Khatib, "A framework for multi-contact multi-body dynamic simulation and haptic display", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2000.

[RKC02] S. Redon, A. Kheddar, and S. Coquillart, "Fast continuous collision detection between rigid bodies", in *Proc. of Eurographics (Computer Graphics Forum)*, 2002.

[RKK97] D.C. Ruspini, K. Kolarov, and O. Khatib, "The haptic display of complex graphical environments", in *Proc. of ACM SIGGRAPH*, 1997, pp. 345–352.

[RPP+01] M. Renz, C. Preusche, M. Pötke, H.-P. Kriegel, and G. Hirzinger, "Stable haptic interaction with virtual environments using an adapted voxmap-pointshell algorithm", *Eurohaptics Conf.*, 2001.

[Rup00] A. H. Rupert. "An instrumentation solution for reducing spatial disorientation mishaps", *IEEE Eng. Medi. Biol.*, Vol. 19, No. 2, pp. 71–80, 2000.

[RV64] I. Rock and J. Victor. "Vision and touch: An experimentally created conflict between the two senses", *Science*, Vol. 143, pp. 594–596, 1964.doi:org/10.1126/science.143.3606.594

[Sal99] J. K. Salisbury. "Making graphics physically tangible", *Communications of the ACM*, Vol. 42, No. 8, pp. 74–81, 1999.

[SBB96] M. A. Srinivasan, G. L. Beauregard, and D. L. Brock, "The impact of visual information on the haptic perception of stiffness in virtual environments", *ASME Winter Annual Meeting*, 1996.

[SBC97] P. Stewart, P. Buttolo, and Y. Chen, "Cad data representations for haptic virtual prototyping", *ASME Design Engineering Technical Conferences*, 1997.

[SBM+95] K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles, "Haptic rendering: Programming touch interaction with virtual objects", in Pat Hanrahan and Jim Winget, eds., *1995 Symp. on Interactive 3D Graphics*, pp. 123–130. ACM SIGGRAPH, New York: April 1995.

[SDS+00] M. R. Sirouspour, S. P. DiMaio, S. E. Salcudean, P. Abolmaesumi, and C. Jones, "Haptic interface control—design issues and experiments with a planar device", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2000.

[Sei90] R. Seidel, "Linear programming and convex hulls made easy", in *Proc. 6th Ann. ACM Conf. on Computational Geometry*, pp. 211–215, Berkeley, California, 1990.

[SFL+06] J. L. Schoner, M. R. Falvo, S. T. Lord, R. M. Taylor, and M. C. Lin, "Interactive simulation of fibrin fibers in virtual environments", in *Proc. of IEEE Virtual Reality Conf.*, 2006.

[Shi92] K. Shimoga, "Finger force and touch feedback issues in dextrous manipulation. *NASA-CIRSSE Int. Conf. on Inetelligent Robotic Systems for Space Exploration*, 1992.

[Sny95] John M. Snyder, "An interactive tool for placing curved surfaces without interpenetration", in Robert Cook, ed., *SIGGRAPH 95 Conf. Proc.*, Annual Conf. Series, pp. 209–218. Reading, MA: Addison Wesley, August 1995.

[SOM04] A. Sud, M. A. Otaduy, and D. Manocha. "DiFi: Fast 3D distance field computation using graphics hardware", *Comput. Graph. Forum*, Vol. 23, No. 3, pp. 557–566, 2004.doi:org/10.1111/j.1467-8659.2004.00787.x

[SP96] J. Siira and D. K. Pai, "Haptic textures—a stochastic approach", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, 1996, pp. 557–562.

[SPD00] C. Spence, F. Pavani, and J. Driver, "Crossmodal links between vision and touch in covert endogenous spatial attention", *J. Exp. Psychol: Hum. Perception Perfor.*, Vol. 26, pp. 1298–1319, 2000.

[SPG03] C. Sigg, R. Peikert, and M. Gross, "Signed distance transform using graphics hardware", in *Proc. of IEEE Visualization*, 2003.

[ST96] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction", *Int. J. Nume. Methods Eng.*, Vol. 39, No. 14, pp. 2673–2691, 1996.

[ST00] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with coulomb friction", *IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 162–169.

[SU93] M. Slater and M. Usoh, "An experimental exploration of presence in virtual environments", Technical Report 689, Department of Computer Science, University College London, 1993.

[Sut65] I. Sutherland, "The ultimate display", *Proc. of IFIP*, 1965, pp. 506–508.

[SV94] S. E. Salcudean and T. D. Vlaar, "On the emulation of stiff walls and static friction with a magnetically levitated input/output device", in *Proc. of ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 1994.

[THM+03] M. Teschner, B. Heidelberger, M. Müeller, D. Pomeranets, and M. Gross, "Optimized spatial hashing for collision detection of deformable objects", in *Proc. of Vision, Modeling and Visualization*, 2003.

[TJC97] T. V. Thompson, D. Johnson, and E. Cohen, "Direct haptic rendering of sculptured models", *Proc. of ACM Symp. on Interactive 3D Graphics*, 1997, pp. 167–176.

[TKH+05] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Furhmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino, "Collision detection for deformable objects", *Comput. Graph. Forum*, Vol. 24, No. 1, pp. 61–81, 2005.

[TRC+93] R. M. Taylor, W. Robinett, V. L. Chii, F. Brooks, and W. Wright, "The nanomanipulator: A virtual-reality interface for a scanning tunneling microscope", in *Proc. of ACM SIGGRAPH*, 1993, pp. 127–134.

[UNB+02] B. J. Unger, A. Nicolaidis, P. J. Berkelman, A. Thompson, S. J. Lederman, R. L. Klatzky, and R. L. Hollis, "Virtual peg-in-hole performance using a 6-dof magnetic levitation haptic device: Comparison with real forces and with visual guidance alone", in *Proc. of Haptics Symp.*, 2002, pp. 263–270.

[van01] G. van den Bergen, "Proximity queries and penetration depth computation on 3d game objects", in *Game Developers Conf.*, 2001.

[WM03] M. Wan and W. A. McNeely, "Quasi-static approximation for 6 degrees-of-freedom haptic rendering", in *Proc. of IEEE Visualization*, 2003, pp. 257–262.

[WS85] P. Walker and S. Smith, "Stroop interference based on the multimodal correlates of haptic size and auditory pitch", *Perception*, Vol. 14, No. 6, pp. 729–736, 1985.

[WS03] S. P. Walker and J. K. Salisbury, "Large haptic topographic maps: Marsview and the proxy graph algorithm", in *Proc. of ACM Symp. on Interactive 3D Graphics*, 2003, pp. 83–92.

[Wu00] D. Wu, "Penalty methods for contact resolution", *Game Developers Conf.*, 2000.

[YSLM04] S. Yoon, B. Salomon, M. C. Lin, and D. Manocha, "Fast collision detection between massive models using dynamic simplification", *Eurographics Symp. on Geometry Processing*, 2004.

[ZS95] C. Zilles and K. Salisbury, "A constraint-based god object method for haptics display", in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robotics and Systems*, 1995.

Biographies

Ming C. Lin is a Professor of Computer Science in the University of North Carolina at Chapel Hill. She received her Ph.D. in Electrical Engineering and Computer Science from University of California at Berkeley. She is recognized worldwide for her research on haptic rendering and real-time physically-based interaction techniques for virtual reality and interactive 3D graphics. She has served as a program and conference chair of many international conferences, as an associate editor and guest editor of several journals in these areas, and as a steering committee member of ACM SIGGRAPH and Eurographics Symposium on Computer Animation and World Haptics Conference. She also co-edited the book, "Applied Computational Geometry".

Miguel A. Otaduy is a post-doctoral research associate at the Computer Graphics Lab at ETH Zurich. He received a Ph.D. in Computer Science from the University of North Carolina at Chapel Hill in 2004, as the result of his work in haptic rendering of geometrically complex objects. He has published in the areas of haptic rendering, collision detection, and physically-based simulation of rigid and deformable objects. He has also served in the program committee of several international conferences, and has taught and organized tutorials on haptic rendering at the ACM SIGGRAPH and Eurographics conferences. He is currently a sub-project leader for the soft-tissue modeling cluster of the Swiss National Center of Competence in Research Co-Me (computational medicine).