Haptic Rendering: Introductory Concepts

Kenneth Salisbury and Francois Conti
Stanford University

Federico Barbagli
Stanford University and University of Siena, Italy

Haptic rendering allows users to "feel" virtual objects in a simulated environment. This article surveys current haptic systems and discusses some basic haptic-rendering algorithms.

In the past decade we've seen an enormous increase in interest in the science of haptics. The quest for better understanding and use of haptic abilities (both human and nonhuman) has manifested itself in heightened activity in disciplines ranging from robotics and telerobotics; to computational geometry and computer graphics; to psychophysics, cognitive science, and the neurosciences.

This issue of *IEEE CG&A* focuses on *haptic rendering*. *Haptics* broadly refers to touch interactions (physical contact) that occur for the purpose of perception or manipulation of objects. These interactions can be between a human hand and a real object; a robot end-effector and a real object; a human hand and a simulated object (via haptic interface devices); or a variety of combinations of human and machine interactions with real, remote, or virtual objects. *Rendering* refers to the process by which desired sensory stimuli are imposed on the user to convey information about a virtual haptic object. At the simplest level, this information is contained in the representation of the object's physical attributes—shape, elasticity, texture, mass, and so on. Just as a sphere visually rendered with simple shading techniques will look different from the same sphere rendered with ray-tracing techniques, a sphere haptically rendered with a simple penalty function will feel different from the same sphere rendered with techniques that also convey mechanical textures and surface friction.

As in the days when people were astonished to see their first wire-frame computer-generated images, people are now astonished to feel their first virtual object. Yet the rendering techniques we use today will someday seem like yesterday's wire-frame displays—the first steps into a vast field.

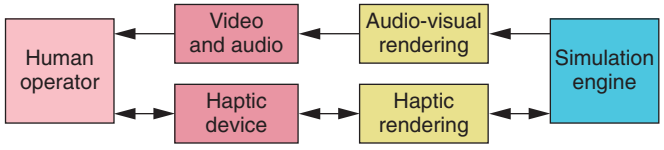
To help readers understand the issues discussed in this issue's theme articles, we briefly survey haptic systems and the techniques needed for rendering the way objects feel. We also discuss basic haptic-rendering algorithms that help us decide what force should be exerted and how we will deliver these forces to users. A sidebar discusses key points in the history of haptics.

Architecture for haptic feedback

Virtual reality (VR) applications strive to simulate real or imaginary scenes with which users can interact and perceive the effects of their actions in real time. Ideally the user interacts with the simulation via all five senses; however, today's typical VR applications rely on a smaller subset, typically vision, hearing, and more recently, touch.

Figure 1 shows the structure of a VR application incorporating visual, auditory, and haptic feedback. The application's main elements are:

- the *simulation engine*, responsible for computing the virtual environment's behavior over time;
- *visual, auditory, and haptic rendering algorithms*, which compute the virtual environment's graphic, sound, and force responses toward the user; and
- *transducers*, which convert visual, audio, and force signals from the computer into a form the operator can perceive.



1 **Basic architecture for a virtual reality application incorporating visual, auditory, and haptic feedback.**

The human operator typically holds or wears the haptic interface

Published by the IEEE Computer Society

History of Haptics

In the early 20th century, psychophysicists introduced the word *haptics* (from the Greek haptesthai meaning *to touch*) to label the subfield of their studies that addressed human touch-based perception and manipulation. In the 1970s and 1980s, significant research efforts in a completely different field—robotics—also began to focus on manipulation and perception by touch. Initially concerned with building autonomous robots, researchers soon found that building a dexterous robotic hand was much more complex and subtle than their initial naive hopes had suggested. In time these two communities—one that sought to understand the human hand and one that aspired to create devices with dexterity inspired by human abilities—found fertile mutual interest in topics such as sensory design and processing, grasp control and manipulation, object representation and haptic information encoding, and grammars for describing physical tasks.

In the early 1990s a new usage of the word *haptics* began to emerge. The confluence of several emerging technologies made virtualized haptics, or *computer haptics*,[1] possible. Much like computer graphics, computer haptics enables the display of simulated objects to humans in an interactive manner. However, computer haptics uses a display technology through which objects can be physically palpated.

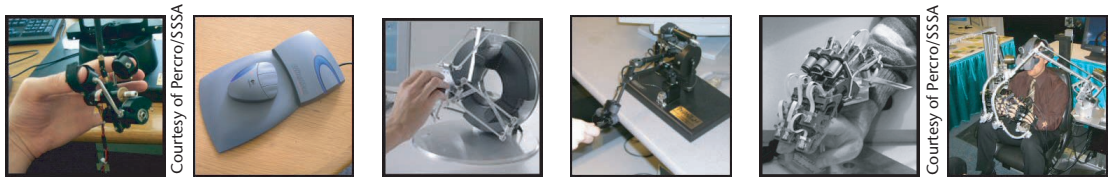
This new sensory display modality presents information by exerting controlled forces on the human hand through a haptic interface (rather than, as in computer graphics, via light from a visual display device). These forces depend on the physics of mechanical contact. The characteristics of interest in these forces depend on the response of the sensors in the human hand and other body parts (rather than on the eye's sensitivity to brightness, color, motion, and so on).

Unlike computer graphics, haptic interaction is bidirectional, with energy and information flows both to and from the user.

Although Knoll demonstrated haptic interaction with simple virtual objects at least as early as the 1960s, only recently was sufficient technology available to make haptic interaction with complex computer-simulated objects possible. The combination of high-performance force-controllable haptic interfaces, computational geometric modeling and collision techniques, cost-effective processing and memory, and an understanding of the perceptual needs of the human haptic system allows us to assemble computer haptic systems that can display objects of sophisticated complexity and behavior. With the commercial availability of 3 degree-of-freedom haptic interfaces, software toolkits from several corporate and academic sources, and several commercial haptics-enabled applications, the field is experiencing rapid and exciting growth.

Reference

1. M.A. Srinivasan and C. Basdogan, "Haptics in Virtual Environments: Taxonomy, Research Status, and Challenges," *Computers and Graphics*, vol. 21, no. 4, 1997, pp. 393-404.



2 Sample of increasingly more complex haptic devices: (a) force-reflecting gripper, (b) Logitech Wingman force-feedback mouse, (c) ForceDimension's Omega haptic device, (d) SensAble's Phantom haptic device, (e) the Hand Force Feedback exoskeleton, and (f) Immersion's Haptic Workstation.

device and perceives audiovisual feedback from audio (computer speakers, headphones, and so on) and visual displays (a computer screen or head-mounted display, for example).

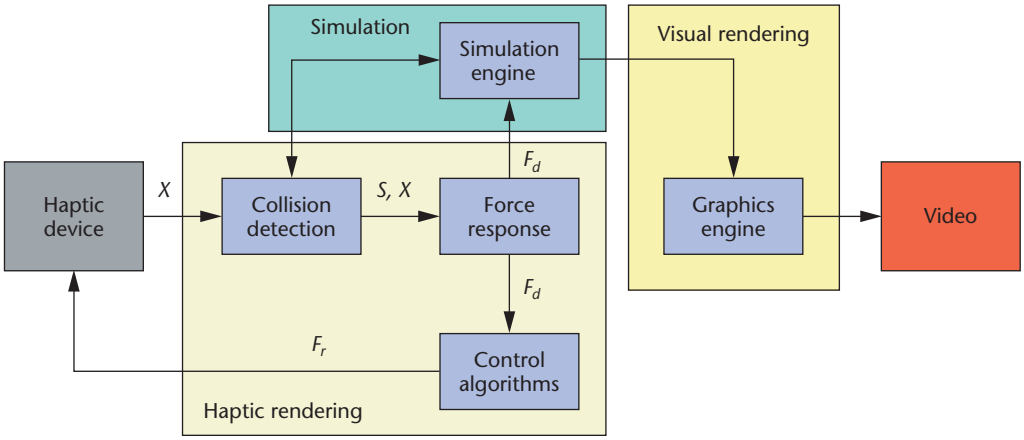
Whereas audio and visual channels feature unidirectional information and energy flow (from the simulation engine toward the user), the haptic modality exchanges information and energy in two directions, from and toward the user. This bidirectionality is often referred to as the single most important feature of the haptic interaction modality.

Haptic interface devices

An understanding of some basic concepts about haptic interface devices will help the reader through the remainder of the text. A more complete description of the elements that make up such systems is available elsewhere.[1]

Haptic interface devices behave like small robots that exchange mechanical energy with a user. We use the term *device-body interface* to highlight the physical connection between operator and device through which energy is exchanged. Although these interfaces can be in contact with any part of the operator's body, hand interfaces have been the most widely used and developed systems to date. Figure 2 shows some example devices.

One way to distinguish between haptic interface devices is by their grounding locations. For interdigit tasks, force-feedback gloves, such as the Hand Force Feedback (HFF),[2] read finger-specific contact information and output finger-specific resistive forces, but can't reproduce object net weight or inertial forces. Similar handheld devices are common in the gaming industry and are built using low-cost vibrotactile transducers, which produce synthesized vibratory effects. Exoskeleton mechanisms or body-based haptic interfaces, which a person wears on the arm or leg, present more complex multiple degree-of-freedom (DOF) motorized devices. Finally, ground-based devices include force-reflecting joysticks and desktop haptic interfaces.



3 We split haptic rendering into three main blocks. *Collision-detection* algorithms provide information about contacts *S* occurring between an avatar at position *X* and objects in the virtual environment. *Force-response* algorithms return the ideal interaction force F_d between avatar and virtual objects. *Control* algorithms return a force F_r to the user, approximating the ideal interaction force to the best of the device's capabilities.

Another distinction between haptic interface devices is their intrinsic mechanical behavior. *Impedance* haptic devices simulate mechanical impedance—they read position and send force. *Admittance* haptic devices simulate mechanical admittance—they read force and send position. Simpler to design and much cheaper to produce, impedance-type architectures are most common. Admittance-based devices, such as the Haptic Master,[3] are generally used for applications requiring high forces in a large workspace.

Haptic interface devices are also classified by the number of DOF of motion or force present at the device-body interface—that is, the number of dimensions characterizing the possible movements or forces exchanged between device and operator. A DOF can be passive or actuated, sensed or not sensed.

Characteristics commonly considered desirable for haptic interface devices include

- low back-drive inertia and friction;
- minimal constraints on motion imposed by the device kinematics so free motion feels free;
- symmetric inertia, friction, stiffness, and resonate-frequency properties (thereby regularizing the device so users don't have to unconsciously compensate for parasitic forces);
- balanced range, resolution, and bandwidth of position sensing and force reflection; and
- proper ergonomics that let the human operator focus when wearing or manipulating the haptic interface as pain, or even discomfort, can distract the user, reducing overall performance.

We consider haptic rendering algorithms applicable to single- and multiple-DOF devices.

System architecture for haptic rendering

Haptic-rendering algorithms compute the correct interaction forces between the haptic interface representation inside the virtual environment and the virtual objects populating the environment. Moreover, haptic-rendering algorithms ensure that the haptic device correctly renders such forces on the human operator.

An *avatar* is the virtual representation of the haptic interface through which the user physically interacts with the virtual environment. Clearly the choice of avatar depends on what's being simulated and on the haptic device's capabilities. The operator controls the avatar's position inside the virtual environment. Contact between the interface avatar and the virtual environment sets off action and reaction forces. The avatar's geometry and the type of contact it supports regulates these forces.

Within a given application the user might choose among different avatars. For example, a surgical tool can be treated as a volumetric object exchanging forces and positions with the user in a 6D space or as a pure point representing the tool's tip, exchanging forces and positions in a 3D space.

Several components compose a typical haptic rendering algorithm. We identify three main blocks, illustrated in Figure 3.

Collision-detection algorithms detect collisions between objects and avatars in the virtual environment and yield information about where, when, and ideally to what extent collisions (penetrations, indentations, contact area, and so on) have occurred.

Force-response algorithms compute the interaction force between avatars and virtual objects when a collision is detected. This force approximates as closely as possible the contact forces that would normally arise during contact between real objects. Force-response algorithms typically operate on the avatars' positions, the positions of all objects in the virtual environment, and the collision state between avatars and virtual objects. Their return values are normally force and torque vectors that are applied at the device-body interface.

Hardware limitations prevent haptic devices from applying the exact force computed by the force-response algorithms to the user. Control algorithms command the haptic device in such a way that minimizes the error between ideal and applicable forces. The discrete-time nature of the haptic-rendering algorithms often makes

this difficult, as we explain later in the article. Desired force and torque vectors computed by force-response algorithms feed the control algorithms. The algorithms' return values are the actual force and torque vectors that will be commanded to the haptic device.

A typical haptic loop consists of the following sequence of events:

- Low-level control algorithms sample the position sensors at the haptic interface device joints.
- These control algorithms combine the information collected from each sensor to obtain the position of the device-body interface in Cartesian space—that is, the avatar's position inside the virtual environment.
- The collision-detection algorithm uses position information to find collisions between objects and avatars and report the resulting degree of penetration or indentation.
- The force-response algorithm computes interaction forces between avatars and virtual objects involved in a collision.
- The force-response algorithm sends interaction forces to the control algorithms, which apply them on the operator through the haptic device while maintaining a stable overall behavior.

The simulation engine then uses the same interaction forces to compute their effect on objects in the virtual environment. Although there are no firm rules about how frequently the algorithms must repeat these computations, a 1-KHz servo rate is common. This rate seems to be a subjectively acceptable compromise permitting presentation of reasonably complex objects with reasonable stiffness. Higher servo rates can provide crisper contact and texture sensations, but only at the expense of reduced scene complexity (or more capable computers).

The following sections explain the basic principles of haptic-rendering algorithms, paying particular attention to force-response algorithms. Although the ability to detect collisions is an important aspect of computing contact force response, given the familiarity of *CG&A*'s readership with the topic, we don't dwell on it here. The geometric problem of efficiently detecting when and where contact and interobject penetrations occur continues to be an important research topic in haptics and related fields. The faster real-time needs of haptic rendering demand more algorithmic performance. One solution is to accept less accuracy and use simpler collision model geometries. Alternately, researchers are adapting graphics-rendering hardware to enable fast real-time collision detection among complex objects. Lin and Manocha give a useful survey of collision-detection algorithms for haptics.[4]

Computing contact-response forces

Humans perceive contact with real objects through sensors (mechanoreceptors) located in their skin, joints, tendons, and muscles. We make a simple distinction between the information these two types of sensors can acquire. *Tactile information* refers to the information acquired through sensors in the skin with particular reference to the spatial distribution of pressure, or more generally, tractions, across the contact area. *Kinesthetic information* refers to the information acquired through the sensors in the joints. Interaction forces are normally perceived through a combination of these two. A tool-based interaction paradigm provides a convenient simplification because the system need only render forces resulting from contact between the tool's avatar and objects in the environment. Thus, haptic interfaces frequently utilize a *tool handle* physical interface for the user.

To provide a haptic simulation experience, we've designed our systems to recreate the contact forces a user would perceive when touching a real object. The haptic interfaces measure the user's position to recognize if and when contacts occur and to collect information needed to determine the correct interaction force. Although determining user motion is easy, determining appropriate display forces is a complex process and a subject of much research. Current haptic technology effectively simulates interaction forces for simple cases, but is limited when tactile feedback is involved.

In this article, we focus our attention on force-response algorithms for rigid objects. Compliant object-response modeling adds a dimension of complexity because of nonnegligible deformations, the potential for self-collision, and the general complexity of modeling potentially large and varying areas of contact.

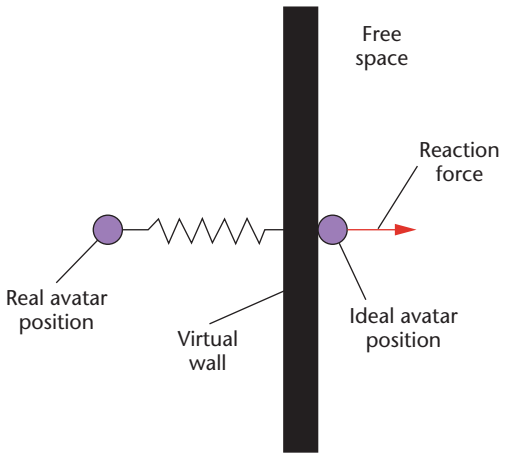
We distinguish between two types of forces: forces due to object geometry and forces due to object surface properties, such as texture and friction.

Geometry-dependent force-rendering algorithms

The first type of force-rendering algorithms aspires to recreate the force interaction a user would feel when touching a frictionless and textureless object. Such interaction forces depend on the geometry of the object being touched, its compliance, and the geometry of the avatar representing the haptic interface inside the virtual environment. Although exceptions exist,[5] the DOF necessary to describe the interaction forces between an avatar and a virtual object typically matches the actuated DOF of the haptic device being used. Thus for simpler devices, such as a 1-DOF force-reflecting gripper (Figure 2a), the avatar consists of a couple of points that can only move and exchange forces along the line connecting them. For this device type, the force-rendering algorithm computes a simple 1-DOF squeeze force between the index finger and the thumb, similar to the force you would feel when cutting an object with scissors. When using a 6-DOF haptic device, the avatar can be an object of any shape. In this case, the force-rendering algorithm computes all the interaction forces between the object and the virtual environment and applies the resultant force and torque vectors to the user through the haptic device.

We group current force-rendering algorithms by the number of DOF necessary to describe the interaction force being rendered.

One-DOF interaction. A 1-DOF device measures the operator's position and applies forces to the operator along one spatial dimension only. Types of 1-DOF interactions include opening a door with a knob that is



4 Virtual wall concept, a 1-DOF interaction. The operator moves and feels forces only along one spatial dimension.

constrained to rotate around one axis, squeezing scissors to cut a piece of paper, or pressing a syringe's piston when injecting a liquid into a patient. A 1-DOF interaction might initially seem limited; however, it can render many interesting and useful effects.

Rendering a *virtual wall*—that is, creating the interaction forces that would arise when contacting an infinitely stiff object—is the prototypical haptic task. As one of the most basic forms of haptic interaction, it often serves as a benchmark in studying haptic stability.[6–8] The discrete-time nature of haptic interaction means that the haptic interface avatar will always penetrate any virtual object. A positive aspect of this is that the force-rendering algorithm can use information on how far the avatar has penetrated the object to the compute interaction force. However, this penetration can cause some unrealistic effects to arise, such as vibrations in the force values, as we discuss later in the article. As Figure 4 illustrates, if we assume the avatar moves along the x-axis and $x < x_W$ describes the wall, the simplest algorithm to render a virtual wall is given by

$$F = \begin{cases} 0 & x > x_W \\ K(x_W - x) & x \le x_W \end{cases}$$

where K represents the wall's stiffness and thus is ideally very large. More interesting effects can be accomplished for 1-DOF interaction.[9,10]

Two-DOF interaction. Examples of 2-DOF interactions exist in everyday life—for example, using a mouse to interact with a PC. Using 2-DOF interfaces to interact with 3D objects is a bit less intuitive. It's possible, however, and is an effective way to interact with simpler 3D virtual environments while limiting the costs and complexity of haptic devices needed to render the interactions. Two-DOF rendering of 3D objects is, in some cases, like pushing a small ball over the surface of a 3D object under the influence of gravity. Various techniques enable this type of rendering by projecting the ideal 3-DOF point-contact interaction force on a plane,[11,12] or by evaluating the height change between two successive contact points on the same surface.[13]

Three-DOF interaction. Arguably one of the most interesting events in haptics' history was the recognition, in the early 1990s, of the point interaction paradigm's usefulness. This geometric simplification of the general 6-DOF problem assumes that we interact with the virtual world with a point probe, and requires that we only compute the three interaction force components at the probe's tip. This greatly simplifies the interface device design and facilitates collision detection and force computation. Yet, even in this seemingly simple case, we find an incredibly rich array of interaction possibilities and the opportunity to address the fundamental elements of haptics unencumbered by excessive geometric and computational complexity.

To compute force interaction with 3D virtual objects, the force-rendering algorithm uses information about how much the probing point, or avatar, has penetrated the object, as in the 1-DOF case. However, for 3-DOF interaction, the force direction isn't trivial as it usually is for 1-DOF interaction.

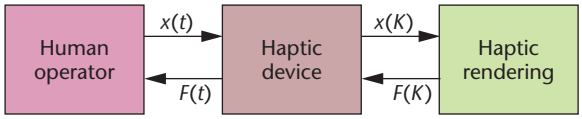
Various approaches for computing force interaction for virtual objects represented by triangular meshes exist. Vector field methods use a one-to-one mapping between position and force. Although these methods often work well, they don't record past avatar positions. This makes it difficult to determine the interaction force's direction when dealing with small or thin objects, such as the interaction with a piece of sheet metal, or objects with complex shapes. Nonzero penetration of avatars inside virtual objects can cause the avatars to cross through such a thin virtual surface before any force response is computed (that is, an undetected collision occurs). To address the problems posed by vector field methods, Zilles and Salisbury and Ruspini et al. independently introduced the god-object[14] and proxy algorithms.[15] Both algorithms are built on the same principle: Although we can't stop avatars from penetrating virtual objects, we can use additional variables to track a physically realistic contact on the object's surface—the god object or proxy. Placing a spring between avatar position and god object/proxy creates a realistic force feedback to the user. In free space, the haptic interface avatar and the god object/proxy are collocated and thus the force response algorithm returns no force to the user. When colliding with a virtual object, the god object/proxy algorithm finds the new god object/proxy position in two steps:

1. It finds a set of active constraints.
2. Starting from its old position, the algorithm identifies the new position as the point on the set of active constraint that is closest to the current avatar position.

Morgenbesser and Srinivasan's introduction of *force shading*—the haptic equivalent of Phong shading—successively refined both algorithms.[16] Whereas graphic-rendering interpolated normals obtain more smooth-looking meshes, haptic-rendering interpolated normals obtain smooth-changing forces throughout an object's surface.

Walker and Salisbury recently proposed an interesting variation of the god-object/proxy algorithms applicable to cases involving triangular meshes based on large quantities of polygons.[17]

Salisbury and Tarr introduced an extension of the god-object algorithm for virtual objects based on implicit surfaces with an analytical representation.[18] For implicit surfaces, collision detection is much faster and we can calculate many of the variables necessary for computing the interaction force, such as its direction and intensity, using closed analytical forms. Other examples of 3-DOF interaction include algorithms for interaction with NURBS-based[19] and with Voxels-based objects.[20]



5 Haptic devices create a closed loop between user and haptic-rendering/simulation algorithms. $x(t)$ and $F(t)$ are continuous-time position and force signals exchanged between user and haptic device. $x(K)$ and $F(K)$ are discrete-time position and force signals exchanged between haptic device and virtual environment.

More than 3-DOF interaction. Although the point interaction metaphor has proven to be surprisingly convincing and useful, it has limitations. Simulating interaction between a tool's tip and a virtual environment means we can't apply torques through the contact. This can lead to unrealistic scenarios, such as a user feeling the shape of a virtual object using the tool's tip while the rest of the tool lies inside the object.

To improve on this situation, some approaches use avatars that enable exertion of forces or torques with more than 3 DOF. Borrowing terminology from the robotic-manipulation community, Barbagli et al.[21] developed an algorithm to simulate 4-DOF interaction through soft-finger contact—that is, a point contact with friction that can support moments (up to a torsional friction limit) about the contact normal. This type of avatar is particularly handy when using multiple-point interaction to grasp and manipulate virtual objects.

Basdogan et al. implemented 5-DOF interaction, such as occurs between a line segment and a virtual object, to approximate contact between long tools and virtual environments.[22] This ray-based rendering technique allows us to simulate the interaction of tools by modeling them as a set of connected line segments and a virtual object.

Several researchers have developed algorithms providing for 6-DOF interaction forces. For example, McNeely et al.[23] simulated interaction between modestly complex rigid objects within an arbitrarily complex environment of static rigid objects represented by voxels, and Otaduy and Lin[24] simulated contact between complex polygonal environments and haptic probes.

Surface property-dependent force-rendering algorithms

All real surfaces contain tiny irregularities or indentations. Obviously, it's impossible to distinguish each irregularity when sliding a finger over an object. However, tactile sensors in the human skin can feel their combined effects when rubbed against a real surface. Although this article doesn't focus on tactile displays, we briefly present the state of the art for algorithms that can render virtual objects' haptic textures and friction properties.

Micro-irregularities act as obstructions when two surfaces slide against each other and generate forces tangential to the surface and opposite to motion. Friction, when viewed at the microscopic level, is a complicated phenomenon. Nevertheless, simple empirical models exist, such as the one Leonardo da Vinci proposed and Charles Augustin de Coulomb later developed in 1785. Such models served as a basis for the simpler frictional models in 3 DOF.[14,15]

Researchers outside the haptic community have developed many models to render friction with higher accuracy—for example, the Karnopp model for modeling stick-slip friction, the Bristle model, and the reset integrator model. Higher accuracy, however, sacrifices speed, a critical factor in real-time applications. Any choice of modeling technique must consider this trade-off. Keeping this trade-off in mind, researchers have developed more accurate haptic-rendering algorithms for friction (see, for instance, Dupont et al.[25]).

A texture or pattern generally covers real surfaces. Researchers have proposed various techniques for rendering the forces that touching such textures generates. Many of these techniques are inspired by analogous techniques in modern computer graphics. In computer graphics, texture mapping adds realism to computer-generated scenes by projecting a bitmap image onto surfaces being rendered. The same can be done haptically. Minsky[11] first proposed haptic texture mapping for 2D; Ruspini et al. later extended his work to 3D scenes.[15] Researchers have also used mathematical functions to create synthetic patterns. Basdogan et al.[22] and Costa and Cutkosky[26] investigated the use of fractals to model natural textures while Siira and Pai[27] used a stochastic approach.

Controlling forces delivered through haptic interfaces

So far we've focused on the algorithms that compute the ideal interaction forces between the haptic interface avatar and the virtual environment. Once such forces have been computed, they must be applied to the user. Limitations of haptic device technology, however, have sometimes made applying the force's exact value as computed by force-rendering algorithms impossible.

Various issues contribute to limiting a haptic device's capability to render a desired force or, more often, a desired impedance. For example, haptic interfaces can only exert forces with limited magnitude and not equally well in all directions, thus rendering algorithms must ensure that no output components saturate, as this would lead to erroneous or discontinuous application of forces to the user.

In addition, haptic devices aren't ideal force transducers. An ideal haptic device would render zero impedance when simulating movement in free space, and any finite impedance when simulating contact with an object featuring such impedance characteristics. The friction, inertia, and backlash present in most haptic devices prevent them from meeting this ideal.

A third issue is that haptic-rendering algorithms oper-

ate in discrete time whereas users operate in continuous time, as Figure 5 illustrates. While moving into and out of a virtual object, the sampled avatar position will always lag behind the avatar's actual continuous-time position. Thus, when pressing on a virtual object, a user needs to perform less work than in reality; when the user releases, however, the virtual object returns more work than its real-world counterpart would have returned. In other terms, touching a virtual object extracts energy from it. This extra energy can cause an unstable response from haptic devices.[7]

Finally, haptic device position sensors have finite resolution. Consequently, attempting to determine where and when contact occurs always results in a quantization error. Although users might not easily perceive this error, it can create stability problems.

All of these issues, well known to practitioners in the field, can limit a haptic application's realism. The first two issues usually depend more on the device mechanics; the latter two depend on the digital nature of VR applications.

As mentioned previously, haptic devices feature a bidirectional flow of energy, creating a feedback loop that includes user, haptic device, and haptic-rendering/simulation algorithms, as Figure 5 shows. This loop can become unstable due to the virtual environment energy leaks.

The problem of stable haptic interaction has received a lot of attention in the past decade. The main problem in studying the loop's stability is the presence of the human operator, whose dynamic behavior can't be generalized with a simple transfer function. Researchers have largely used passivity theory to create robust algorithms that work for any user.

For a virtual wall such as the one in Figure 4, Colgate and Brown[6] analytically showed that a relation exists between the maximum stiffness a device can render, the device's level of mechanical damping, the level of digital damping commanded to the device, and the servo rate controlling the device.[6] More specifically, to have stable interaction, the relationship $b > KT/2 + B$ should hold. That is, the device damping b should always be higher than the sum of the level of digital damping that can be controlled to the device B and the product $KT/2$ where K is the stiffness to be rendered by the device and T is the servo rate period. Stiffer walls tend to become unstable for higher servo rate periods, resulting in high-frequency vibrations and possibly uncontrollably high levels of force. Increasing the level of mechanical damping featured by the device can limit instability, even though this limits the device's capabilities of simulating null impedance when simulating the device's free-space movements. Thus high servo rates (or low servo rate periods) are a key issue for stable haptic interaction.

Two main sets of techniques for limiting unstable behavior in haptic devices exist. The first set includes solutions that use virtual damping to limit the energy flow from the virtual environment toward the user when it could create unstable behavior.[8,28] Colgate introduced *virtual coupling*, a connection between haptic device and virtual avatar consisting of stiffness and damping, which effectively limits the maximum impedance that the haptic display must exhibit.[28] A virtual coupling lets users create virtual environments featuring unlimited stiffness levels, as the haptic device will always attempt to render only the maximum level set by the virtual coupling. Although this ensures stability, it doesn't make a haptic device stably render higher stiffness levels.

The second set of techniques includes solutions that attempt to speed up haptic servo rates by decoupling force-response algorithms from other slower algorithms, such as collision-detection, visual-rendering, and virtual environment dynamics algorithms.[29] This can be accomplished by running all of these algorithms in different threads with different servo rates, and letting the user interact with a simpler local virtual object representation at the highest possible rate that can be accomplished on the system.

Four main threads exist. The *visual-rendering loop* is typically run at rates of up to 30 Hz. The *simulation thread* is run as fast as possible congruent with the simulated scene's overall complexity. A *collision-detection thread*, which computes a local representation of the part of the virtual object closest to the user avatar, is run at slower rates to limit CPU usage. Finally a faster *collision detection and force response* is run at high servo rates. An extremely simple local representation makes this possible (typical examples include planes or spheres). Surface discontinuities are normally not perceived, given that the maximum speed of human movements is limited and thus the local representation can always catch up with the current avatar position. This approach has gained success in recent years with the advent of surgical simulators employing haptic devices, because algorithms to accurately compute deformable object dynamics are still fairly slow and not very scalable.[30,31]

Conclusion

As haptics moves beyond the buzzes and thumps of today's video games, technology will enable increasingly believable and complex physical interaction with virtual or remote objects. Already haptically enabled commercial products let designers sculpt digital clay figures to rapidly produce new product geometry, museum goers feel previously inaccessible artifacts, and doctors train for simple procedures without endangering patients.

Past technological advances that permitted recording, encoding, storage, transmission, editing, and ultimately synthesis of images and sound profoundly affected society. A wide range of human activities, including communication, education, art, entertainment, commerce, and science, were forever changed when we learned to capture, manipulate, and create sensory stimuli nearly indistinguishable from reality. It's not unreasonable to expect that future advancements in haptics will have equally deep effects. Though the field is still in its infancy, hints of vast, unexplored intellectual and commercial territory add excitement and energy to a growing number of conferences, courses, product releases, and invention efforts.

For the field to move beyond today's state of the art,

researchers must surmount a number of commercial and technological barriers. Device and software tool-oriented corporate efforts have provided the tools we need to step out of the laboratory, yet we need new business models. For example, can we create haptic content and authoring tools that will make the technology broadly attractive? Can the interface devices be made practical and inexpensive enough to make them widely accessible?

Once we move beyond single-point force-only interactions with rigid objects, we should explore several technical and scientific avenues. Multipoint, multihand, and multiperson interaction scenarios all offer enticingly rich interactivity. Adding submodality stimulation such as tactile (pressure distribution) display and vibration could add subtle and important richness to the experience. Modeling compliant objects, such as for surgical simulation and training, presents many challenging problems to enable realistic deformations, arbitrary collisions, and topological changes caused by cutting and joining actions.

Improved accuracy and richness in object modeling and haptic rendering will require advances in our understanding of how to represent and render psychophysically and cognitively germane attributes of objects, as well as algorithms and perhaps specialty hardware (such as haptic or physics engines) to perform real-time computations.

Development of multimodal workstations that provide haptic, visual, and auditory engagement will offer opportunities for more integrated interactions. We're only beginning to understand the psychophysical and cognitive details needed to enable successful multimodality interactions. For example, how do we encode and render an object so there is a seamless consistency and congruence across sensory modalities—that is, does it look like it feels? Are the object's density, compliance, motion, and appearance familiar and unconsciously consistent with context? Are sensory events predictable enough that we consider objects to be persistent, and can we make correct inference about properties?

Finally, we shouldn't forget that touch and physical interaction are among the fundamental ways in which we come to understand our world and to effect changes in it. This is true on a developmental as well as an evolutionary level. For early primates to survive in a physical world, Frank Wilson suggested that

> a new physics would eventually have to come into this their brain, a new way of registering and representing the behavior of objects moving and changing under the control of the hand. It is precisely such a representational system—a syntax of cause and effect, of stories, and of experiments, each having a beginning, a middle, and an end—that one finds at the deepest levels of the organization of human language.[32]

Our efforts to communicate information by rendering how objects feel through haptic technology, and the excitement in our pursuit, might reflect a deeper desire to speak with an inner, physically based language that has yet to be given a true voice. ■

References

1. V. Hayward and O. Astley, "Performance Measures for Haptic Interfaces," *Proc. 1996 Robotics Research: 7th Int'l Symp.*, E. Giralt and G. Hirzinger, eds., Springer Verlag, 1996, pp. 195-207.
2. M. Bergamasco, "Manipulation and Exploration of Virtual Objects," *Artificial Life and Virtual Reality*, N. Magnenat Thalmann and D. Thalmann, eds., John Wiley & Sons, 1994, pp. 149-160.
3. R.Q. VanderLinde et al., "The Hapticmaster, a New High-Performance Haptic Interface," *Proc. Eurohaptics*, Edinburgh Univ., 2002, pp. 1-5.
4. M.C. Lin and D. Manocha, "Collision and Proximity Queries," *Handbook of Discrete and Computational Geometry*, 2nd ed., J. O'Rourke and E. Goodman, eds., CRC Press, 2004.
5. F. Barbagli and K. Salisbury, "The Effect of Sensor/Actuator Asymmetries in Haptic Interfaces," *Proc. 11th Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*, IEEE CS Press, 2003, pp. 140-147.
6. J. Colgate and J. Brown, "Factors Affecting the Width of a Haptic Display," *Proc. IEEE Int'l Conf. Robotics and Automation* (ICRA 94), IEEE CS Press, 1994, pp. 3205-3210.
7. B. Gillespie and M. Cutkosky, "Stable User-Specific Haptic Rendering of the Virtual Wall," *Proc. ASME Int'l Mechanical Eng. Conf. and Exposition*, vol. 58, ASME, 1996, pp. 397-406.
8. R. Adams and B. Hannaford, "Stable Haptic Interaction with Virtual Environments," *IEEE Trans. Robotics and Automation*, vol. 15, no. 3, 1999, pp. 465-474.
9. S. Snibbe et al., "Haptic Techniques for Media Control," *Proc. 14th Ann. ACM Symp. User Interface Software and Technology* (UIST 01), vol. 3, no. 2, ACM Press, 2001, pp. 199-208.
10. A.M. Okamura et al., "The Haptic Scissors: Cutting in Virtual Environments," *Proc. IEEE Int'l Conf. Robotics and Automation* (ICRA 03), vol. 1, IEEE CS Press, 2003, pp. 828-833.
11. M. Minsky, *Computational Haptics: The Sandpaper System for Synthesizing Texture for a Force Feedback Display*, doctoral dissertation, Mass. Inst. of Technology, 1995.
12. Y. Shi and D.K. Pai, "Haptic Display of Visual Images," *Proc. IEEE Virtual Reality Ann. Int'l Symp.* (VRAIS 97), IEEE CS Press, 1997, pp. 188-191.
13. V. Hayward and D. Yi, "Change of Height: An Approach to the Haptic Display of Shape and Texture Without Surface Normal," *Experimental Robotics VIII*, Springer Tract in Advanced Robotics 5, B. Siciliano and P. Dario, eds., Springer-Verlag, 2003, pp. 570-579.
14. C. Zilles and J.K. Salisbury, "A Constraint-Based God-Object Method for Haptic Display," *Proc. IEE/RSJ Int'l Conf. Intelligent Robots and Systems, Human Robot Interaction, and Cooperative Robots*, vol. 3, IEEE CS Press, 1995, pp. 146-151.
15. D.C. Ruspini, K. Kolarov, and O. Khatib, "The Haptic Display of Complex Graphical Environments," *Proc. ACM Siggraph*, ACM Press, 1997, pp. 345-352.
16. H.B. Morgenbesser and M.A. Srinivasan, "Force Shading for Haptic Perception," *Proc. ASME Dynamic Systems and*

Control Division, vol. 58, ASME, 1996, pp. 407-412; http://touchlab.mit.edu/publications/1996_003.pdf.
17. S. Walker and J.K. Salisbury, "Large Haptic Topographic Maps: Marsview and the Proxy Graph Algorithm," *Proc. ACM Symp. Interactive 3D*, ACM Press, 2003, pp. 83-92.
18. K. Salisbury and C. Tarr, "Haptic Rendering of Surfaces Defined by Implicit Functions," *ASME Dynamic Systems and Control Division*, vol. 61, ASME, 1997, pp. 61-67.
19. T.V. Thompson et al., "Maneuverable Nurbs Models Within a Haptic Virtual Environment," *Proc. 6th Ann. Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*, vol. 61, IEEE CS Press, 1997, pp. 37-44.
20. R.S. Avila and L.M. Sobierajski, "A Haptic Interaction Method for Volume Visualization," *Proc. IEEE Visualization*, IEEE CS Press, 1996, pp. 197-204.
21. F. Barbagli, K. Salisbury, and R. Devengenzo, "Enabling Multifinger, Multihand Virtualized Grasping," *Proc. IEEE Int'l Conf. Robotics and Automation* (ICRA 03), vol. 1, IEEE CS Press, 2003, pp. 806-815.
22. 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," *Proc. ASME Dynamic Systems and Control Division*, vol. 61, ASME, 1997, pp. 77-84; http://touchlab.mit.edu/publications/1997_001.pdf.
23. W. McNeely, K. Puterbaugh, and J. Troy, "Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling," *Proc. ACM Siggraph*, ACM Press, 1999, pp. 401-408.
24. M.A. Otaduy and M. Lin, "Sensation Preserving Simplification for Haptic Rendering," *Proc. ACM Siggraph,* ACM Press, 2003, pp. 543-553.
25. V. Hayward and B. Armstrong, "A New Computational Model of Friction Applied to Haptic Rendering," *Experimental Robotics VI*, P. Corke and J. Trevelyan, eds., LNCIS 250, Springer-Verlag, 2000, pp. 403-412.
26. M. Costa and M. Cutkosky, "Roughness Perception of Haptically Displayed Fractal Surfaces," *Proc. ASME Dynamic Systems and Control Division*, vol. 69, no. 2, 2000, pp. 1073-1079.
27. J. Siira and D. Pai, "Haptic Textures: A Stochastic Approach," *Proc. IEEE Int'l Conf. Robotics and Automation* (ICRA 96), IEEE CS Press, 1996, pp. 557-562.
28. J. Colgate, M. Stanley, and J. Brown, "Issues in the Haptic Display of Tool Use," *Proc. Int'l Conf. Intelligent Robots and Systems*, IEEE CS Press, 1995, pp. 140-145.
29. Y. Adachi, T. Kumano, and K. Ogino, "Intermediate Representation for Stiff Virtual Objects," *Proc. IEEE Virtual Reality Ann. Int'l Symp.*, IEEE CS Press, 1995, pp. 203-210.
30. F. Barbagli, D. Prattichizzo, and K. Salisbury, "Dynamic Local Models for Stable Multicontact Haptic Interaction with Deformable Objects," *Proc. Haptics Symp*, 2003, pp. 109-116.
31. M. Mahvash and V. Hayward, "Passivity-Based High-Fidelity Haptic Rendering of Contact," *Proc. IEEE Int'l Conf. Robotics and Automation* (ICRA 03), IEEE CS Press, 2003, pp. 3722-3728.
32. F. Wilson, *The Hand: How Its Use Shapes the Brain, Language, and Human Culture*, Vintage Books, 1999.

Kenneth Salisbury *is a professor in the computer science and surgery departments at Stanford University. His research interests include human-machine interaction, collaborative computer-mediated haptics, and surgical simulation. Salisbury has a PhD in mechanical engineering from Stanford. He has served on the US National Science Foundation's Advisory Council for Robotics and Human Augmentation, as scientific advisor to Intuitive Surgical Inc., and as technical advisor to Robotic Ventures Inc.*

Francois Conti *is pursuing a PhD in the field of soft tissue modeling and simulation for real-time applications at the Stanford University Robotics Laboratory. His research interests include algorithms to simulate deformable objects and haptic interface design. Conti has an MS in electrical engineering from the Swiss Federal Institute of Technology in Lausanne (EPFL).*

Federico Barbagli *is an assistant professor in the school of engineering at the University of Siena, Italy, and a research fellow at the Stanford University Robotics Laboratory. His research interests include haptic rendering algorithms, haptic control, and haptic device design. Barbagli has a PhD in robotics and control from Scuola Superiore S.Anna, Italy.*

Readers may contact Kenneth Salisbury at the Stanford Robotics Lab, CS Dept., Gates building, 353 Serra Mall, Stanford, CA 94305; jks@robotics.stanford.edu.