High-Fidelity Haptic Synthesis of Contact with Deformable Bodies

Mohsen Mahvash
Real Contact Inc.

Vincent Hayward
McGill University

A method for synthesizing the haptic response of nonlinear deformable objects from data obtained by offline simulation helps create surgical simulators with high-fidelity haptic feedback.

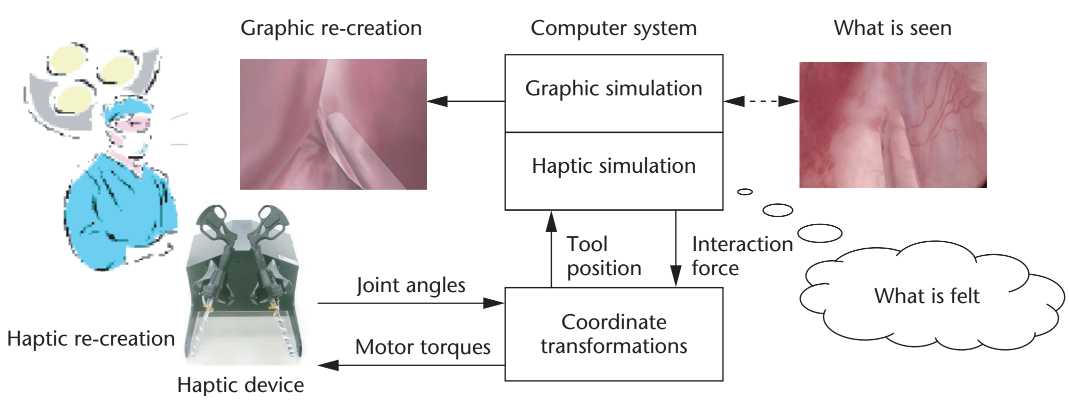
Haptic displays provide users with artificially created tactile sensations. One important use of these displays is to recreate the experience caused by contact between a tool and an object. This capability can be useful in several applications, such as surgical simulators, because users experience an enhanced sense of realism when a haptic simulation is combined with a graphic simulation. Haptic displays require two essential subsystems: a haptic device, which typically has a handle connected to sensors and actuators, and a computational system that interfaces with the device.

As shown in Figure 1, the haptic simulator's function is to reproduce what is felt. Because the physics of light differs from the physics of mechanical interactions, we must recognize that although graphic and haptic simulations can share the encoding of certain properties, such as shape, they must differ in many other aspects, such as models, mathematical techniques, and implementation. One key difference is the need to synthesize a force vector for haptics, as opposed to an illumination field for graphics.

If all the components of the simulator are sufficiently accurate, the system can generate a vivid experience of interacting with real objects. The process of computing and generating forces in response to the interaction with virtual objects is sometimes called haptic rendering.[1]

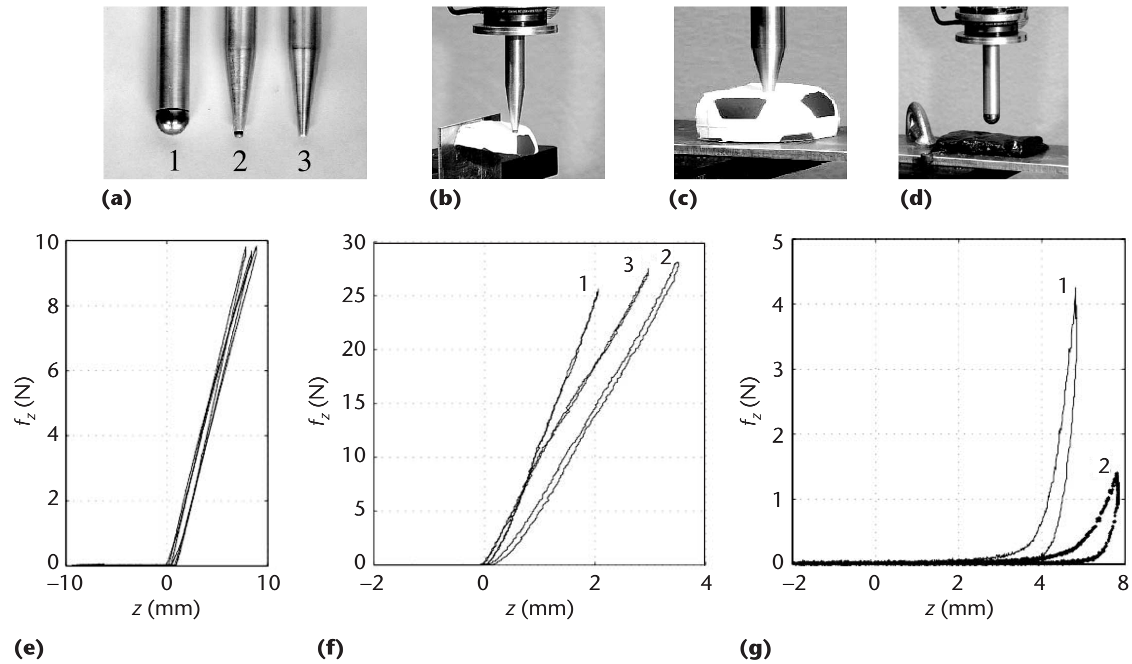
For many applications, high fidelity is important. This article proposes a definition for high-fidelity haptic synthesis in terms of four requirements:

- resemblance of virtual force responses with actual responses,
- force continuity under all allowed maneuvers,
- passivity of the virtual environment, and
- high-force update rate to combat the adverse effects of discretization.



1 Immersion's Laparoscopic Surgical workstation is an example of a device for use in surgical training simulators. A complete system comprises two software components, one for haptics and one for graphics each driving specific interfacing hardware. General-purpose devices for high-quality 3D interaction with virtual environments include Sensable's Phantom or MPBT's Freedom-6S, which is based on direct-drive technology and returns forces and torques. (Image courtesy of Danny Grant.)

Published by the IEEE Computer Society



2 **(a) We used three tools to perform standard tests under various conditions: tool with a large spherical tip, tool with a small spherical tip, and flat punch. (b) Test of a loosely supported rubber sample with all three tools. (c) Same sample with good support. (d) Liver sample test. (e) Responses from b differed only by small details. (f) Responses were similar for the flat punch and for the large sphere, although of dissimilar sizes. Response for the small sphere is completely different. (g) Effects of nonlinear behavior of liver sample were dramatic.**

To gain these properties in a haptic simulation, we start from the notion of deflection, which we define as the displacement of the initial point of contact between an instrument and a deformable body.

No matter how complicated the underlying continuum mechanics, the force response due to deformation is a function of deflection only, even if the deformable body is anisotropic, inhomogeneous, nonlinear elastic, and if large deformation occurs. In other words, the full specification of the mutual response of any pair of objects is equivalent to describing a different force field for each pair of surface points. This notion yields an exceedingly practical method to realize nonlinear haptic synthesis, which is very efficient for many applications. In this article, we treat the case of the interaction of a tool (a surgical instrument) with something deformable (an organ consisting of soft and hard tissues).

Simulating soft objects

For a virtual environment made of rigid objects, haptic rendering should at least represent the location of these objects and their geometry, texture, and frictional properties. When the virtual objects are deformable, such as is the case of most organs, haptic rendering must account for more properties: materials, support, and internal structure. In terms of the surgical example, there will be dramatic differences in the response of the same tissue sample depending on how it's attached to a bone or other organs, or whether it's prestretched or limp. Similarly, a bladder will respond completely differently if it's empty or filled. Or if you take a section of homogenous tissue, a hard inclusion embedded in it (such as a cyst) or a softer section (such as a diseased area) will cause a different feel.

The majority of previous approaches have assumed that contact interactions between a virtual tool and a virtual object occur at one, or at a small number of points. The limitations introduced by this assumption can be readily appreciated by considering that during contact there is always *localized deformation* via a contact surface. This causes the force of contact and the feel to depend critically on the details of the shape of the objects in contact.

A simple experiment illustrates this idea. If you take a well-supported piece of material like a foam mouse pad laid flat on a table, a sharp point produces a small force for a given indentation. A flat punch, however, will produce a force that could be orders of magnitude larger for the same indentation. It's also due to the properties of foams, which, like tissues, are cellular structures having distinct deformation regimes corresponding to microscopic structural changes that yield large-scale effects.

There are two main cases that arise as a result of St. Venant's principle, which states the general conditions when the effects of localized deformation can be neglected. A consequence of this principle in a haptic simulation is as follows. If a deformable body is loosely supported, one is permitted to ignore the contact details, but if it is well supported, the contact details dominate. This is because the small-scale features of the tool's shape have a huge impact on the localized deformation (see Figure 2).[2]

In our tests with a loosely supported rubber sample (see Figure 2b), the three responses differed only by small details because global deformation hid the details of local deformation (see Figure 2e). When we tested the same sample with good support (see Figure 2c), as seen in Figure 2f, the responses were similar for the flat punch and for the large sphere, at least during initial indentation, although they had dissimilar sizes. The response of the small sphere, however, was completely different, especially in the initial stages of deformation. We did the same test with a sample of fresh liver (see Figure 2d). This time, the effects of the nonlinear behavior of the tissue were dramatic, as shown in Figure 2g.

Even microscopic changes in a tool shape can make huge differences in the response—as is evident when trying to cut meat with a dull knife. The shape of the tool and the way it makes contact with the body is therefore a key aspect of an interaction that haptic rendering should account for. Moreover, ignoring the shape of the tool and the contact area between the tool and the body can lead to artifacts caused by the discrete nature of computer representations—haptic clicks and pops. Thus, for deformable bodies, a point contact representation for realistic virtual interactions with deformable bodies is an idealization that is neither necessary nor sufficient.[2,3]

High fidelity implies that the essential aspects of an environment are represented with the fewest defects possible. To firm up this vague statement, we can list the properties that a simulation should share with the physical world. We believe that four of them are necessary:

- The simulation should synthesize force responses that reproduce the responses of actual interactions, not some arbitrary responses of an idealized point contact.
- Just as in the physical word, the simulation should compute force responses that are continuously related to displacement. Objects always contact each other by surfaces that grow and shrink as contacts are made and undone. Continuity is a property not easily obtained from computer representations that are, by necessity, discrete.
- The simulation should preserve the passivity of objects, which implies that the principle of energy conservation should be obeyed by the simulation the same way it is obeyed by the physical world.
- The update rate of the synthesis should be high enough to ensure that time discretization does not remove any important aspects of the virtual interaction, nor does it add any artifacts such as limit cycles or aliased signals.

With these points in mind, we can offer an overview of an approach to haptic rendering that is physically correct and provides these requirements. This approach forms the basis of a software system under development, called HapticEngine.

Response synthesis

An interesting way to think of haptic rendering is to compare it to teleoperation. In teleoperation, two robots, a master and a slave, are connected to each other. The master arm—a device similar to a haptic interface—continuously measures the position of a handle maneuvered by the operator and reproduces forces experienced by the slave arm. The slave arm continuously tracks the position of the master so when its end-effector comes into contact with its environment, operators experience the interaction force as if they were holding the slave arm.

In haptic rendering, a computer simulation replaces both the slave arm's end-effector and its environment to create synthetic interaction responses. One common method to create realistic responses is to encode the virtual object's geometry, material properties, and boundary conditions, and to predict the contact responses by solving the continuum equations of deformation. The finite element method and its variants is a highly developed method for doing so. One major problem when using this method is the number of computations required; this number can be enormous and unpredictable unless you make some major simplifications.

Researchers have explored various approaches to speed up the numerical solution of the continuum equations. We divide these approaches into two groups.

In the first group, high-order dynamic deformation models represent the body, and then time integration help solve them efficiently in real time with explicit integration, with fixed meshing for large deformation,[4] or with adaptive meshing for non-Hookean materials.[5] For particle-based methods, implicit integration was suggested,[6] and adaptive multirate integration with multispatial resolution was proposed.[7] In essence, to make the high computation rate for forces possible, most of these approaches represent the continuum medium by a dense network of lumped mass-spring elements and therefore must grossly over-simplify the contact problem.

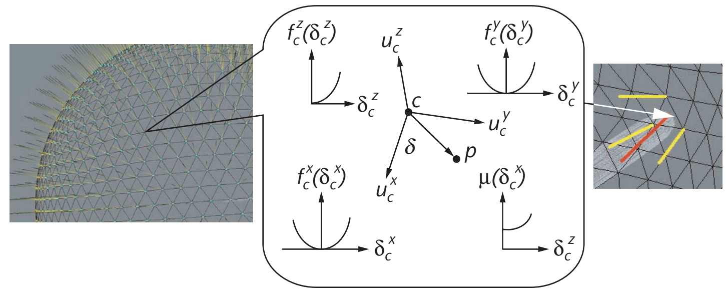
In the second group, the approach is to precalculate a large number of responses in offline processing, looking them up in real time to synthesize interaction responses.[8-12] In these methods, a set of algebraic linear relations among the nodal quantities at the free boundary are derived from a linear model so that the deformation responses given by unit forces applied to each node of the free boundary can be calculated during a preprocessing phase. The deformation response of the boundary can be calculated as the superposition of responses of each nodal force in a preset contact region. Precalculation methods are effective but, unfortunately, cannot represent nonlinear aspects such as large deformation or nonlinear elasticity.

In all these cases, due to discretization, an accurate computation of the force deflection response requires virtually unbounded numbers of elements to represent a body in the contact region that can be small, even microscopically small, yielding prohibitively large computational loads.

We have developed a precalculation method for nonlinear contact problems. This method handles nonlinear deformation arising from large deformation, nonlinear elasticity, or inhomogeneity and anisotropy, by means of massive precomputation of the force responses.[2,13] With this approach, the number of precomputation steps is proportional to the number of possible cases of interaction. In principle, this number increases with the product of the number of sample points on the body by the number of sample points on the tool. It also grows with the number of dimensions considered.

Fortunately, in practice, we don't have to consider all cases. We can reduce the precomputation burden by computing forces for coarse sample sets and then resampling a fine mesh structure for online processing. The storage requirements are also acceptable. Say we represent a complex multi-axis nonlinear response with 100 bytes using polynomial interpolators. An instrument interacting in 10 different nonlinear manners requires 1 Kbyte of storage per node. One Gbyte of RAM would allow us to store one million nodes representing a gigantic one square meter operating field with a different response each millimeter.

We now illustrate this method for the case of an instrument coming into sticking and sliding contact with deformable objects at several possible places—for instance, in laparoscopic or endoscopic procedures. In a precalculation step, we first obtain a finite number of force responses for contacts between the instrument tool and the body within the range and maneuverability.



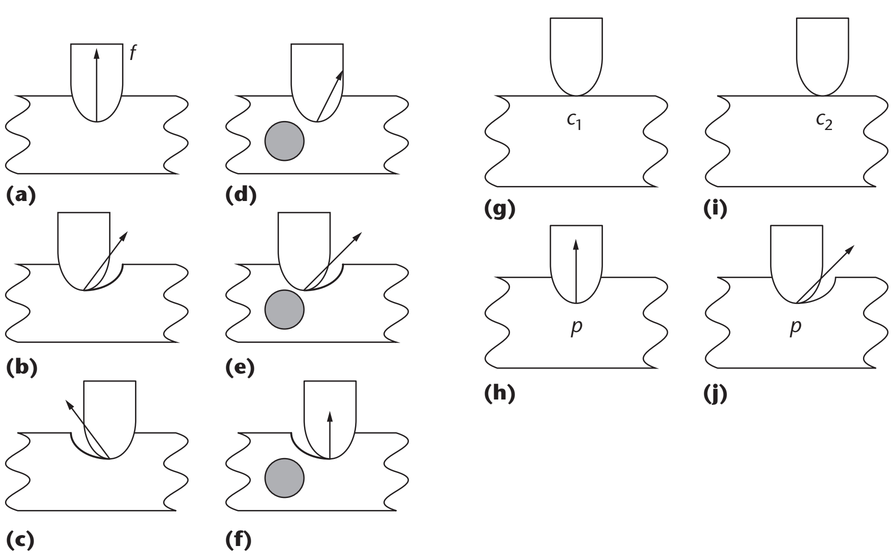
3 A simple pointed tool interacts with a rubber ball. We record a finite number of multidimensional response samples on the surface of the undeformed body. These are used to reconstruct the original continuum for any generic initial contact point *c*. We developed a method to reconstruct the nonlinear force field anywhere (shown in red) from neighboring ones (shown in yellow).

Local force field continuum

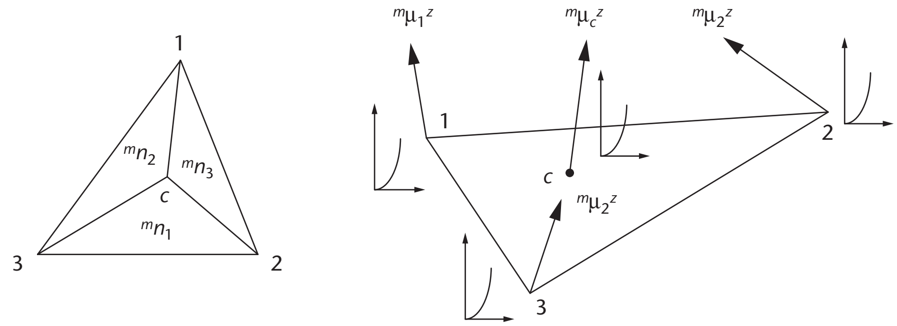
A rather counterintuitive aspect of our method comes from the fact that you must only know the responses at the surface of the *undeformed* body where a contact begins. There is no need to compute its shape during subsequent deformation—at least in the case of a single tool, as shown in Figure 3. This is because a contact must *always* begin in a sticking state. It follows that the initial contact point between a rigid object and an undeformed body is the only quantity required to entirely determine the subsequent response until sliding occurs. We call the set of forces associated with this point a "local force field at c for p" where c is the initial contact point on the body surface and p is the corresponding point on the tool surface.

From a contact mechanics viewpoint, we can view c as the point around which a sticking contact area develops. With no rolling or sliding during deformation, c remains invariant on the undeformed surface of the body. To select the correct force response, we also need p, which is the position of the initial contact over the tool surface. Think, for example, of poking meat with a pointed knife either by the sharp edge, the point, or the dull edge. The points c and $p(t)$ coincide at the time of initial contact, but, subsequently, $p(t)$ changes according to the movements of the tool. Sliding a tool over a body means that a continuum of local force fields must be generated according to the trajectory $c(t)$ of point c.

A local coordinate system $U_c = \{u_c^x, u_c^y, u_c^z\}$ must be defined to encode the response at c. The choice of local coordinates is crucial for a good approximation when encoding a given nonlinear local force field. We can select these local coordinates so that the stored response fits an actual response more accurately and economically, as shown in Figure 4. Define deflection $\delta(t)$ as the difference between $p(t)$ and $c(t)$, $\delta(t) = p(t) - c(t)$. Given U_c, we encode the force response as a function of deflection components $f_c(\delta) = f_c^x(\delta_c^x)u_c^x + f_c^y(\delta_c^y)u_c^y + f_c^z(\delta_c^z)u_c^z$ where the force-deflection curves $f_c^x(\delta_c^x)$, $f_c^y(\delta_c^y)$, and $f_c^z(\delta_c^z)$ can be as complicated, nonlinear, and skewed as dictated by the mechanics of a particular contact situation. According to the desired accuracy, representations can range from linear interpolators to high-order piece-wise polynomial interpolators.



4 Blunt tool indenting soft tissue. (a) A homogenous sample responds in a simple fashion to normal indentation and (b, c) in an axis-symmetric fashion to tangential deflections. (d, e, f) An inhomogenous sample gives a skewed response for the same deflections. (g, h) The importance of encoding whole response fields and not just force responses at a point is shown by a homogenous material sample indented with normal deflection at initial contact point c_1 and deflected to point *p*. (i, j) The same tool deflecting the same sample to the same point but with a different initial contact point c_2 returns a different force due to a different deflection.



5 Interpolation weights $^m n_l$ are associated to a simplex patch *m* and vertices $l = 1, 2, 3$. Natural coordinates are based on the areas $^m A$ of the three simplices associated to patch *m* defined the three vertices and by an interior point *c*. A typical patch size of 10 mm is sufficient for most applications, but the patch size also depends on the desired level of accuracy.

In our synthesis method, we don't have to assume that the undeformed body's surface is fixed. To simulate interaction with a moving organ, such as a beating heart or a heaving diaphragm, we still evaluate $\delta(t) = p(t) - c(t)$, but $c(t)$ represents a point of initial contact on a time varying boundary.

We call the determination of the response at a point a *test*, and therefore call the known response at some points *c* and *p*, a *test point pair*. For the sake of generality, we can obtain responses using three possibly combined methods. Precalculation was already mentioned. Also, actual tests with real tools and real bodies are possible. In some rare cases, we can obtain analytical solutions from the techniques of contact mechanics.

Reconstruction of the continuum

Long ago, it was observed by Dahl that all contacts are compliant and that friction results from the finite amount of elastic energy that a contact can store. In other words, when the tangential deflection of a contact exceeds a limiting value, the contact undergoes inelastic deformation to limit deflection, resulting in energy dissipation. The relevance of this model for haptic synthesis is discussed elsewhere.[14] For deformable bodies, the model applies equally well but at a scale much greater than with hard objects.

For every contact between a tool and a body, there exists a coefficient of friction μ that relates the limiting tangential deflection to the normal deflection δ^z via $f^r_c(\delta^r_s) = \mu_c(\delta^z_c) f^z_c(\delta^z_c)$, where f^r_c and δ^r_c are the respective projections of f_c and δ_c on the surface of the undeformed body. To simulate friction during sliding, *c* moves over the body surface such that $|\delta^r| \le \delta^r_s$ at all times. We add the coefficient $\mu_c(\cdot)$ to the response specification at *c*. A frictionless contact is simply the special case of a null tangential response.

The process described here applies to the synthesis of a single force field for a pair of points, but while sliding a tool over the surface of a body, the local force field will vary according to $c(t)$ and $p(t)$. As in computer graphics, we need an interpolation method to generate all possible responses from a finite set. We represent the haptic responses by functions, which must be interpolated. Because the responses are vector functions, the process requires two steps: one to interpolate the coordinate bases and a second to interpolate the components.

For now, consider the case of a tool always contacting an object in the same manner. We first need to locate a point uniquely on the surface of the object. A convenient and well-known technique for this is based on so-called natural coordinates, also variously known as simplex coordinates, or normalized barycentric coordinates. We divide the surface into triangular patches and associate each patch to an index *m* having three vertices indexed by $l = 1, 2, 3$, as shown in Figure 5. A point *c* is located within a patch *m* by three numbers $^m n_l(c)$. We use these numbers to smoothly interpolate the nodal coordinate systems into a new coordinate system at *c*. In a second step, the nodal response curves are also interpolated at *c*. This generates a local force field for any point initial contact point *c* on the surface of the body. In essence, from the knowledge of the force fields at the vertices, we can reconstruct a continuum of force fields.

$$^m n_l(c) = \frac{^m A_l(c)}{^m A}$$

$$u^x_c = \sum_l {}^m n_l(c) {}^m u^x_l$$
$$u^y_c = \sum_l {}^m n_l(c) {}^m u^y_l$$
$$u^z_c = \sum_l {}^m n_l(c) {}^m u^z_l$$

and

$$f^x_c(\delta^x) = \sum_l {}^m n_l(c) {}^m f^x_l(\delta^x)$$
$$f^y_c(\delta^y) = \sum_l {}^m n_l(c) {}^m f^y_l(\delta^y)$$
$$f^z_c(\delta^z) = \sum_l {}^m n_l(c) {}^m f^z_l(\delta^z)$$

If we have only one local force field at each test point, we can only represent contacts that are always performed in the same manner by the same tool. For example, it could be a pointed tool used to interact with the pointed end. Realistically, we should allow for the same tool to interact with a body in different ways. A good example is that of the same pointed tool but one that can be maneuvered to interact with the body using the stem as well. In this case we will have a different family of local fields for each manner with which the tool can interact with the body. For a cylindrical pointed tool, there would be one family for contacts with the tip and another for contacts with the stem. This scheme generalizes to any tool shape.

Arbitrary tool contacts

For arbitrary tool contacts, we perform the online reconstruction of the continuum by interpolating force fields that are elements of the Cartesian product of two sets of initial contact points: one on the body (large) and one on the tool (small). While interference detection between two moving bodies and the determination of the initial contact point c are important and difficult problems, we must skip their discussion and assume these problems solved.

When c belongs to patch j of the body and to patch m of the tool, we label the contact attributes as follows: The unit vectors of the local coordinates are ${}^{jm}u^x_{il}, {}^{jm}u^y_{il}, {}^{jm}u^z_{il}$, where i and l are the indices of the vertices of patches j and m respectively. We note the precalculated force deflection curves ${}^{jm}f^x_{il}(\cdot), {}^{jm}f^y_{il}(\cdot), {}^{jm}f^z_{il}(\cdot)$. We first calculate the local coordinates at point c by a double interpolation of the unit vectors at vertices of the triangles:

$$u^x_c = \sum_i {}^j n_i(c)\left[\sum_l {}^m n_l(c)\, {}^{jm}u^x_{il}\right]$$
$$u^y_c = \sum_i {}^j n_i(c)\left[\sum_l {}^m n_l(c)\, {}^{jm}u^y_{il}\right]$$
$$u^z_c = \sum_i {}^j n_i(c)\left[\sum_l {}^m n_l(c)\, {}^{jm}u^z_{il}\right]$$

and then a second double interpolation determines the force-deflection curves at point c:

$$f^x_c(\delta^x) = \sum_i {}^j n_i(c)\left[\sum_l {}^m n_l(c)\, {}^{jm}f^x_{il}(\delta^x)\right]$$
$$f^y_c(\delta^y) = \sum_i {}^j n_i(c)\left[\sum_l {}^m n_l(c)\, {}^{jm}f^y_{il}(\delta^y)\right]$$
$$f^z_c(\delta^z) = \sum_i {}^j n_i(c)\left[\sum_l {}^m n_l(c)\, {}^{jm}f^z_{il}(\delta^z)\right]$$

This process involves a small amount of computations even if the responses are complicated. The coefficient of friction is also interpolated using an analogous formula.

Passivity

In normal life, we commonly contact passive objects, such as the chair you are probably sitting on at this moment. By a passive object, we mean an object like the chair that doesn't generate energy by itself. If we monitor the energy that flows in and out of the object, it never returns more energy than the energy that was stored in it. Passivity is a property of real environments that should be replicated by artificial force-reflecting environments.

If forces are computed so that they represent a passive object, although the haptic device has actuators, then the whole haptic simulation system will represent passive objects as well. To see this, we state that the computer should always provide a force trajectory $f(t)$ so that the energy at the interaction point never becomes larger than the initial energy at that same point. We call $v(t)$ the velocity trajectory of the point of interaction and write

$$\int_0^t f(\tau)^{\mathrm{T}} \upsilon(\tau)\, d\tau \geq 0$$

The actuators convert a signal into another but always in the same manner. Therefore, they appear as constant factors in the above integral—not changing its sign. Lack of passivity for virtual environments is sufficient to permit the creation of limit cycles while interacting with them.

Transitions between local force fields

Passivity is connected to the notion of static conservative fields. If a static force field is conservative, then the work computed by a line integral along any given path remains independent from this path and only depends on the start and the end points. If our synthesis generates static conservative force fields, then the energy at any interaction point is:

$$E(x(t)) = \int_{t_0}^{t} f(x(\tau))^{\mathrm{T}} \upsilon(\tau)\, d\tau \geq 0 \qquad \text{(static)}$$
$$= \int_{x_0}^{x(t)} f(y)^{\mathrm{T}}\, dy \qquad \text{(conservative)}$$

If the synthetic force is generated by a sequence of locally defined passive fields each activated during time intervals i = 0, 1, … , n—where a time interval corresponds to time during which a particular local field is active—then the total rendering will preserve passivity if the following energy condition holds: $E_i(x) \leq E_{i-1}(x)$.[15] This means that the energy at the interaction point should not increase when there is a transition from the field $(i - 1)$ to the field i. Certain common cases discussed later in the article can violate this condition.

Passivity doesn't depend on the tool being in contact with objects at only one place at a time. Multiple contacts correspond to the activation of multiple fields. This happens, for example, when inserting an instrument between the lobes of an organ. The superposition of independent static conservative fields is still static conservative and hence passivity is preserved.

Time discretization effects

Even if a virtual environment is nominally passive, time discretization will unavoidably add a measure of activity with the time lag between the sampling of position and the physical generation of force by the device.[16,17] This typically results in buzzing behaviors that plague many attempts at haptic simulations. Time discretization will also cause aliasing of force signal whenever the bandwidth of the force model is greater than a half of the rate of force update. This can happen when the simulated body is textured and when the user moves quickly over such simulated surfaces, causing creation of low-frequency artifacts.

Moreover, a digital system usually holds the computed force values for the entire duration of an update period to convert discrete samples into an analog signal. The resulting staircase signal contains high-frequency components that must be filtered out, a job ordinarily left to the amplifiers, the actuators, and the device itself. A low update rate can generate low-frequency harmonics that the system doesn't filter, especially when using direct-drive high-fidelity devices.

A high rate of force update can solve these problems for a wide range of cases. The passivity margin directly relates to the update rate of the simulation.[16,18] A high rate can make the whole interaction passive because it's a fact of physics that devices must exhibit some dissipation, however small. We can show that if the update rate is high enough, the lack of passivity will be compensated. A high rate update also increases the frequency of aliased harmonics of the generated forces so that the device can better mechanically filter them, allowing a wider bandwidth of force response. However, attempting to run all haptic simulations at a high rate clearly is a great limitation. For this reason, we introduced a family of multirate approaches to force generation that preserves the properties we have discussed.

Multirate design for passivity

In our system, two processes run concurrently, possibly on the same machine or on several interconnected ones. The first process runs at a high rate and performs a reduced amount of computations. It essentially evaluates the interpolation equations outlined earlier and supplies forces to the device. The role of the other process is to supply local data to the high rate process. The data associated with a single active patch processed at a high rate lets our system carry out unpredictable and time-consuming operations, such as interference detection at a low rate.

We developed several strategies to guarantee force continuity and passivity at the moment of patch update, although the two processes are essentially asynchronous. Neighboring patches always share two vertices, therefore it's easy to show that the natural area coordinates provide continuity automatically. Passivity is more involved, as we must ensure that the energy condition is always satisfied. Here are some important cases.

We always accomplish the computational detection of interference between a tool and a body with a finite error due to lag in the system and movement of the tool. For a speed of 0.1 meters per second, a 10-ms lag yields a 1-mm error. Left uncompensated, when the high rate process activates the interfering patch, the tool has already penetrated the object, causing a nonzero initial deflection of value δ_0. To compensate for this, the high rate process renders $f_c(\delta - \delta_0)$ for all subsequent local force fields as long as the tool remains in contact with the object, sticking or sliding over it, so $E_i = 0$ at activation time. In essence, the simulated object is moved by a small amount each time a collision occurs. This type of position adjustment is of no consequence on the haptic perception of objects.[19]

Another case arises when you replace a patch by another with some lag while sliding. To ensure energy match at the time of the update, you must force the contact point to remain briefly on the edge of the old patch until a new one is ready to take over. This adds a small error to the tangential deflection but ensures that the energy condition holds.

There are other instances in which the condition could be violated. For example, consider a frictionless sliding movement on the surface of a nonhomogenous body with constant penetration. If the material properties change and become harder during movement, then the energy can increase. If the force field is nominally passive for a sticking contact, a small amount of synthetic friction when transitioning to a sliding contact is normally sufficient to compensate for such possible unphysical energy creation.[15]

Implementation

We are applying the methods described here to create a high-fidelity surgical simulator prototype. The particular surgical procedure involves using a long cylindrical instrument inserted through an orifice to perform the ablation of an organ. We described the undeformed organ geometry and that of the surrounding tissues by triangular meshes loaded from an ordinary VRML file.

Another file in a proprietary format contains the response information as well as information describing the local coordinate frames. An interesting and particularly useful feature of the software allows us to modify the force response information online while interacting with the virtual organ. It is also capable of handling multiple simultaneous contacts made by one instrument.

We organized the application around several threads:

- A hard real-time thread that updates device forces at a high rate (1 or 2 KHz or more according to the needs of the device used).
- A thread in Java that performs interference detection and determines which regions of the body and tool are in contact.
- Several graphics and user-interaction threads written in Java 3D and for the Java virtual machine.

This prototype demonstrates that high-fidelity haptic synthesis is possible even in the case of complicated nonlinear and inhomogenous objects supported in realistic ways, while using ordinary computing platforms.

Future work

The continuum reconstruction approach enables the creation of high-fidelity haptic simulations when using instruments for poking, pulling, sliding, and insertion into an orifice or between preloaded parts of a virtual organ, including the reproduction of many nonlinear effects of interest to surgical simulation. However, by principle, this approach is limited to the simulation of contact conditions that have been precomputed, even sparsely. In essence, this corresponds to all the cases where there are no permanent changes made to a virtual body. These permanent changes can arise principally from simulating two phenomena: plasticity and damage.

From a haptic-simulation viewpoint, we characterize plasticity by the fact that the surface of a body assumes a shape that depends on the occurrence of deflection extrema. Therefore, its simulation requires introducing additional internal states, not only to encode changes in the responses, but also in the shape of the deformable body. Precomputation-interpolation approaches can still accommodates theses cases, but at the expense of more storage.

The distinctive nature of damage, on the other hand, corresponds to creating entirely new surfaces, such as cutting planes or exposure of new surfaces via delamination—the process by which an orange is peeled. In the present state of development, the precalculation-interpolation approach can only deal with cutting along planes or surfaces of dissection that are known in advance. This technique is nevertheless applicable to many cases of practical interest, such as teaching known surgical procedures or rehearsing patient-specific procedures.

We are working toward developing a theory to facilitate the automatic determination of sample density and their dimensionality, given a particular simulation problem. It's possible that function bases other than polynomial approximations could yield more compact encodings. Extending the precomputation-interpolation approach to graphical simulation is possible as well, and is a subject of our continuing work. ■

Acknowledgments

This work was sponsored by the Institute for Robotics and Intelligent Systems and by the Natural Sciences and Engineering Council of Canada.

References

1. K. Salisbury et al., "Haptic Rendering: Programming Touch Interaction with Virtual Objects," *Proc. Symp. Interactive 3D Graphics*, ACM Press, 1995, pp. 123-130.
2. M. Mahvash, V. Hayward, and J.E. Lloyd, "Haptic Rendering of Tool Contact," *Proc. Eurohaptics 2002*, Edinburgh University, pp. 110-115.
3. M. Mahvash and V. Hayward, "Haptic Rendering of Cutting, A Fracture Mechanics Approach," *Haptics-e, The Electronic J. Haptics Research*, vol. 2, no. 3, 2001.
4. X. Wu et al., "Adaptive Nonlinear Finite Elements for Deformable Body Simulation Using Dynamic Progressive Meshes," *Proc. Eurographics*, vol. 20, no. 3, 2001, pp. 349-58.
5. Y. Zhuang and J. Canny, "Haptic Interaction with Global Deformations," *Proc. IEEE Int'l Conf. Robotics and Automation*, 2000. pp. 2428-2433.
6. D. D'Aulignac, R. Balaniuk, and C. Laugier, "A Haptic Interface for a Virtual Exam of the Human Thigh," *Proc. IEEE Int'l Conf. Robotics and Automation*, 2000, pp. 2452-2457.
7. G. Debunne et al., "Dynamic Real-Time Deformations Using Space and Time Adaptive Sampling," *Proc. SIGGRAPH, Computer Graphics and Interactive Techniques*, ACM Press, 2001, pp. 31-36.
8. O. Astley and V. Hayward, "Multirate Haptic Simulation Achieved by Coupling Finite Element Meshes Through Norton Equivalents," *Proc. Int'l Conf. Robotics and Automation,* vol. 2, IEEE Press, 1998, pp. 989-994.
9. S. Cotin, H. Delingette, and N. Ayache, "Real-Time Elastic Deformations of Soft Tissues for Surgery Simulation," *IEEE Trans. Visualization and Computer Graphics*, vol. 5, no. 1, 1999, pp. 62-73.
10. D.L. James and D.K. Pai, "A Unified Treatment of Elastostatic and Rigid Contact Simulation for Real Time Haptics," *Haptics-e, The Electronic J. Haptics Research*, vol. 2, no. 1, 2001.
11. M. Bro-Nielsen, "Finite Element Modeling in Surgery Simulation," *Proc. IEEE*, vol. 86, no. 3, 1998, pp. 490-503.
12. F. Mazzella, K. Montgomery, and J.-C. Latombe, "The Forcegrid: A Buffer Structure for Haptic Interaction with Virtual Elastic Objects," *Proc. IEEE Int. Conf. on Robotics and Automation*, IEEE, 2002, pp. 939-946.
13. M. Mahvash and V. Hayward, "Haptic Simulation of a Tool in Contact with a Nonlinear Deformable Body," *IS4TM: Int'l Symp. Surgery Simulation and Soft Tissue Modelling*, Springer Verlag, 2003, pp. 311-320.
14. V. Hayward and B. Armstrong, "A New Computational Model of Friction Applied to Haptic Rendering," *Experimental Robotics VI*, Springer-Verlag, 2000, pp. 403-412.
15. M. Mahvash and V. Hayward, "Passivity-Based High-Fidelity Haptic Rendering of Contact," *Proc. IEEE Int'l Conf. Robotics and Automation*, 2000, pp. 3722-3728.
16. R.E. Ellis, N. Sarkar, M.A. Jenkins, "Numerical Methods for the Force Reflection of Contact," *ASME Trans. Dynamic Systems, Modeling, and Control*, vol. 119, no. 4, 1997, pp. 768-774.
17. B. Hannaford and J.-H. Ryu, "Stable Haptic Interaction with Virtual Environments," *IEEE Trans. Robotics and Automation*, vol. 18, no. 1, 2002, pp. 1-10.
18. J.E. Colgate and G.G. Schenkel, "Passivity of a Class of Sampled-Data Systems: Application to Haptic Interfaces," *J. Robotic Systems,* vol. 14, no. 1, 1997, pp. 37-47.
19. C.B. Zilles and J.K. Salisbury, "A Constraint-Based God Object Method for Haptic Display," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* IEEE, vol. 3, 1995, pp. 146-151

Mohsen Mahvash *is cofounder and president of Real Contact Inc. His research interests include surgical simulation, physics-based virtual reality, and control systems. Mahvash received a PhD in electrical and computer engineering from McGill University, Montreal.*

Vincent Hayward *is an associate professor of electrical and computer engineering at McGill University, director of the McGill Center for Intelligent Machines, and head of the Haptics Laboratory at McGill. His research interests include haptic displays, haptic perception, robotics, and control systems. Hayward received a PhD in computer science from the University of Paris.*

Readers may contact the authors at the Center for Intelligent Machines, McGill Univ., 3480 University Street, Montreal, Quebec, H3A 2A7, Canada; {mahvash, hayward}@cim.mcgill.ca.