Haptic Interaction with Global Deformations*

Yan Zhuang† John Canny‡
Computer Science Department
University of California, Berkeley, CA 94720-1776

Abstract

Force feedback coupled with a real-time physically realistic graphic display provides a human operator with an artificial sense of presence in a virtual environment. Furthermore, it allows a human operator to interact with the virtual environment through "touch". In this paper, we describe a haptic simulation system that allows a human operator to perform real-time interaction with soft 3D objects that go through large global deformations. We model and simulate such a global deformation using *geometrically nonlinear* finite element methods (FEM). We also introduce an efficient method that computes the force feedback, in real-time, by simulating the collision between the virtual "*proxy*" and the deformable object. To perceptually satisfy a human operator, haptics requires a much higher update frequency (at least 1000Hz) than graphics. We update the graphics using full simulation and interpolate the fully simulated states at a higher frequency to render haptics. The interpolation is made possible by intentionally delaying the display (both graphics and haptics) by one full simulation cycle.

1 Introduction

The word *haptic* refers to something that is associated with the sense of touch. In a haptic simulation, to achieve a virtual sense of touch, the human operator interacts with an active mechanical device, called a *haptic display*. A haptic simulation system includes the following essential elements: a human operator, a haptic display, a graphic display and a virtual environment. The human operator makes physical contact with the haptic display. The coupling of real-time graphic and haptic displays provides the human operator an artificial sense of kinesthetic presence in a virtual environment. Furthermore, it allows a human operator to interact with the virtual environment through "touch".

A haptic display can take on many forms, most commonly a robotic manipulator with the ability to exert forces on a human. One of the most successful haptic displays is the Phantom. Other haptic displays include Salisbury Hand, mini-WAM, Shah finger, etc. [26].

Applications of haptic simulation include, but are not limited to, surgical training, physical rehabilitation, computer-aided design, and entertainment. A haptic simulation system can also enhance a human operator's ability to perform certain tasks [7, 16].

In this paper, instead of addressing a specific application of haptic simulation, we address the bottleneck problem of real-time interaction with large global deformations of 3D soft objects, with physically realistic force feedback. By *global deformations*, we mean deformations, such as twisting and bending of an object, which involve the entire body, in contrast to poking and squeezing, which involves a relatively small region of the deformable object.

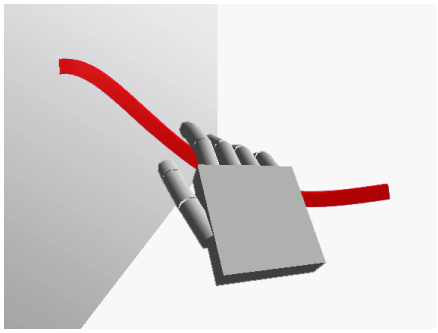

Figure 1: A virtual hand interacting with a soft cantilever beam.

To simplify the control of the haptic device, we represent our haptic device by a virtual *proxy* (section 3). The force feedback exerted on the human operator by the haptic display is simulated by the collision between this virtual proxy and the deformable object (section 5.1). We model and simulate the global deformations of 3D objects using a displacement based *nonlinear finite element method* (FEM) (section 4).

While real-time graphic display requires an update rate of only 30Hz, stable haptic display requires an update rate of at least 1000Hz. In this paper, we describe a simple interpolation scheme (section 5.2) that can interpolate force feedback at the required high frequency, while the virtual environment is only simulated at a lower frequency.

2 Related Work

Our work involves both real-time realistic visual effects and haptic effects. Computer graphics and haptics share the same goal of evoking the sensation of objects by appropriate sensory stimulation. Graphic rendering techniques seek to provide the perception of an object's color, geometry, surface texture, etc., by rendering an appropriate image. Haptic rendering techniques seek to provide the human operator with the appropriate force feedback to "feel" the geometry, surface and material property of the object.

In the computer graphics domain, our work of modeling and simulating a deformable object falls into the realm of physically based modeling. Witkin *et al*[34] summarizes the methods and principles of physically based modeling, which has emerged as an important new approach to computer animation and computer graphics modeling.

*Supported by a Multi-Disciplinary Research Initiative grant for 3D Visualization, sponsored by BMDO with support from ONR.

†yzhuang@cs.berkeley.edu
‡jfc@cs.berkeley.edu

In general, there are two different approaches to modeling deformable objects: the mass spring model and the finite element model. Gibson and Mirtich [8] gives a comprehensive review of this subject.

The mass spring model has had good success in creating visually satisfactory animations. Waters [32] uses a spring model to create a realistic 3D facial expression. Provot *et al*[19] describes a 2D model for animating cloth, using double cross springs. Promayon *et al*[18] presents a mass-spring model of 3D deformable objects and develops some control techniques.

Despite the success in some animation applications, the mass spring models do not model the underlying physics accurately, which makes it unsuitable for simulations that require more accuracy. The structure of the mass spring is often application dependent and hard to interpret. The animation results often vary dramatically with different spring structures. The distribution of the mass to nodes is somewhat (if not completely) arbitrary. Despite its inaccuracy, it does not have visual distortion and it is computationally cheap to integrate over time because the system is, by its very nature, a set of independent algebraic equations, which requires no matrix inversions to solve.

As an alternative, finite element methods (*FEM*) model the continuum much more accurately and their underlying mathematics are well studied and developed. Another similar method is the finite difference method, which is less accurate but simple and appropriate for some applications. Indeed a linear finite difference method over a uniform mesh is just a special case of FEM. Its accuracy and mathematical rigorousness make FEM a better choice for applications such as surgical simulations.

Terzopoulos *et al*[29, 28, 30] applies both finite difference and finite element methods in modeling elastically deformable objects. Celniker *et al*[15] applies FEM to generate primitives that build continuous deformable shapes designed to support a new free-form modeling paradigm. Pieper *et al*[17] applies FEM to computer-aided plastic surgery. Chen [5] animates human muscle using a 20 node hexahedral FEM mesh. Keeve *et al*[10] develops a static anatomy-based facial tissue model for surgical simulation using the FEM. Most recently, Cotin *et al*[6] presents real-time elastic deformation of soft tissues for surgery simulation, which only simulates static deformations.

James and Pai [9] model real time static local deformations using the boundary element method (BEM). BEM has the advantage of solving a smaller system because it only deals with degrees of freedom on the surface of the model. However, the resulting system is dense. Furthermore, a boundary element method cannot be applied to model non-homogeneous material.

Our work differs from the previous work by either one or all of the following: (1) we simulate large global deformations instead of small local deformations; (2) we simulate the dynamic behavior of soft objects rather than the static deformation.

On haptic displays, Salisbury [26] reviews the history of haptic devices. Srinivasan and Salisbury [27] reviews the issues and challenges in haptic feedback. Mark *et al*[11] describes solutions of adding force feedback for static models into computer graphics systems. Adachi *et al*[1] addresses the problem of haptic display of curved surfaces using an intermediate representation. Velula and Baraff [31] discuss the integration of force feedback into their rigid body dynamics simulation system [2]. Minsky *et al*[12] addresses various haptic feedback techniques for surface textures. Ruspini *et al*[24, 25, 23, 21, 22] applies robotic motion planning techniques to haptic interactions in a virtual environment. Furthermore, they describe a new haptic rendering library HL, which enables graphics programmers to add haptics into a graphic virtual environment.

3 Haptic Model Overview

The haptic simulation includes a human operator, a haptic device (such as a PHANToM manipulator, a CyberGrasp glove, etc.), a graphic display, and a virtual environment. The human operator makes *physical* contact with the haptic device through pushing, grasping or some other mechanism. The haptic device provides the operator with a kinesthetic sense of presence in the virtual environment through appropriate force feedback.

In our paper, we describe a system that allows users to virtually interact with objects exhibiting large deformations. The real-time haptic feedback is coupled with a real-time graphic display.

To simplify the control of the haptic device, we simulate the force feedback using a virtual *proxy* similar to that of Ruspini *et al*[25, 23], and the "god object" of Zilles and Salisbury [37]. Our virtual proxy is different from that of Ruspini *et al*[25, 23] because our proxy's motion is guided by the dynamic simulation. Namely upon collision, the motion of the proxy is guided by physics instead of a local minimization.

In section 4, we describe how we simulate the physically realistic global deformation in real-time, using geometrically nonlinear FEM. In section 5, we describe how the simulation can be used to provide haptic feedback to the human operator through the haptic device. In section 5.2, we describe how we display graphics and haptics at different frequencies.

4 Nonlinear Elasticity with FEM

By *global deformations*, we mean deformations that are large and involve the entire body, such as high amplitude bending and twisting (figure 2 and 3). These types of deformation often occur to soft objects, such as tissue in surgical simulations.

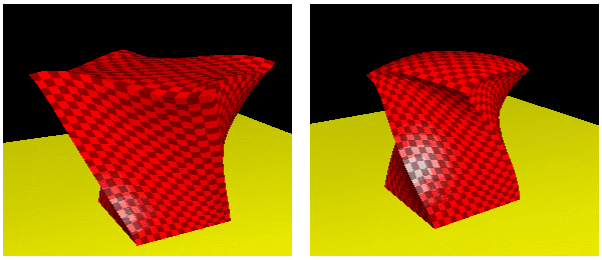

Figure 2: The bottom of the object is fixed and its top is twisted. The top in the left image is distorted (grown bigger) because it is simulated using linear elasticity. The right image shows that the same distortion does not occur with nonlinear elasticity.

The theory of elasticity is a fundamental discipline in studying continuum material. It consists of a consistent set of differential equations that uniquely describe the state of stress, strain and displacement of each point within an elastic deformable body. It consists of equilibrium equations relating the stresses; kinematics equations relating the strains and displacements, constitutive equations relating the stresses and strains; and boundary conditions relating to the physical domain. The theory was first developed by Louis-Marie-Henri Navier, Dimon-Denis Poisson and George Green in the first half of the 19th century [33].

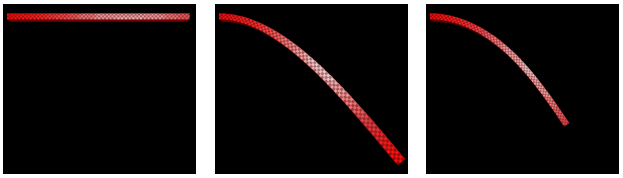

Figure 3: The left image shows a beam at its initial configuration with a fixed left end and a free right end. The middle image shows the *distorted* deformation under gravity, using linear strain. The right image shows the *undistorted* deformation, under the same gravitational force, using quadratic strain (equation (5) and (6)).

Synthesizing those equations allows us to establish a relationship between the deformation of the object and the exerted forces. However an analytic expression of such relationship is impossible, except for a small number of simple problems. *Finite element methods (FEM)* are one way to solve such a set of differential equations. From now on, we will discuss elasticity within the context of finite element methods.

When the geometry of the deformable object is complicated, it is impossible to obtain an analytic solution of an elastic deformation. FEM solves this problem by subdividing the object into small sub-domains with simple shapes (tetrahedra, hexahedra, etc.), called finite elements. The sub-division (mesh) does not only approximate the original geometry, but also leads to a discrete representation of the deformation.

In particular, we apply a *displacement based* finite element method to simulate such deformation. Namely displacements at vertices of the mesh, called nodes, will be calculated. The values at other points within the element are interpolated by continuous functions, usually low order polynomials, using the nodal values. The global equations (the relationship between all the nodal values) are obtained by assembling elementwise equations by imposing inter-element continuity of the solution and balancing of inter-element forces.[1] This essentially requires solving the following system of differential equations:

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{R}(\mathbf{u}) = \mathbf{F} \tag{1}$$

where $\mathbf{u}$ is the $3n$-dimensional nodal displacement vector; $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$, the respective velocity and acceleration vectors; $\mathbf{F}$, the external force vector; $\mathbf{M}$, the $3n \times 3n$ mass matrix; $\mathbf{D}$, the damping matrix; and $\mathbf{R}(\mathbf{u})$, the internal force vectors due to deformation. n is the number of nodes in the FEM model [36].

To our best knowledge the published research ([17, 5, 10, 6]) assume small deformations in their virtual environment. The most simulated deformations are those caused by squeezing and poking at a relatively small surface region. The small deformation assumption leads to the often used linear elasticity model, which is based on the following linear strain approximations:

$$\epsilon_x = \frac{\partial u}{\partial x} \tag{2}$$

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \tag{3}$$

where x, y and z are the independent variables of the cartesian frame, and u, v and w are the corresponding displacement variables at the given point. Other terms of the strain at point (x, y, z), ϵ_y, ϵ_z, γ_{yz} and γ_{zx}, are defined similarly.

This linear strain makes the internal force vector linear with respect to nodal displacement vector. Namely it simplifies equation (1) to the following *linear* system:

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{F} \tag{4}$$

This allows a preprocessing step that computes the constant stiffness matrix $\mathbf{K}$ and its LU factorization. This preprocessing step has been the key to real-time performance in previous works, such as [6], which animates deformations using a sequence of static equilibria.

The problem with this linear strain approximation is that it does not model finite rotation correctly. As a result, it introduces distortions when large global deformations occur (figure 2 and 3), because global deformations usually involve finite rotation of part of the object relative to the rest of it.

To further illustrate this distortion, let us subject an undeformed object to a rigid body rotation. Apparently, the rotation should not introduce any deformation to the object. Namely the strain at any point within the object should be zero. However equations (2) and (3) give a nonzero strain. This "artificial" strain leads to distortion, because the body has to deform in a certain way to balance the stress caused by such an "artificial strain".

To avoid the distortion, as shown in figures (2) and (3), we model the deformation using the exact strain, which is quadratic as following:

$$\epsilon_x = \frac{\partial u}{\partial x} + \frac{1}{2}\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial w}{\partial x}\right)^2\right] \tag{5}$$

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} + \left[\frac{\partial u}{\partial x}\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\frac{\partial v}{\partial y} + \frac{\partial w}{\partial x}\frac{\partial w}{\partial y}\right] \tag{6}$$

The other 4 terms of the strain are defined similarly. It is easy to verify that the above nonlinear strain handles arbitrary large rigid body motions correctly. Namely no artificial strain will be introduced when we subject the object to a rigid body motion.

This quadratic strain makes (1) a nonlinear system, in which the internal force $\mathbf{R}(\mathbf{u})$ is no longer a linear term of nodal displacements. If we solve this nonlinear system using an implicit integration scheme such as [4], real time simulation is impossible for reasonably large meshes.

We observe that a soft material such as live tissue has small stiffness in all directions (not necessarily isotropic). This makes explicit time integration schemes appropriate because we can take large time steps. We apply the explicit Newmark scheme to equation (1), which leads to the following equations:

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \dot{\mathbf{u}}_n \triangle t_n + \frac{1}{2}\ddot{\mathbf{u}}_n \triangle t_n^2 \tag{7}$$

$$(\mathbf{M} + \frac{1}{2}\triangle t_n \mathbf{D})\ddot{\mathbf{u}}_{n+1} = \mathbf{F}_{n+1} - \mathbf{R}(\mathbf{u}_{n+1}) - \mathbf{D}(\dot{\mathbf{u}}_n + \frac{1}{2}\ddot{\mathbf{u}}_n \triangle t_n) \tag{8}$$

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + \frac{1}{2}(\ddot{\mathbf{u}}_n + \ddot{\mathbf{u}}_{n+1})\triangle t_n \tag{9}$$

The order of updating is (7), (8) and then (9). The bottleneck is equation (8), which is a nonlinear system of equations. If we apply a general method, such as Newton's method, to solve this

[1] Detailed discussions of FEM can be found in [20, 35].

equation, it requires inverting a large sparse matrix $\mathbf{M} + \frac{1}{2}\triangle t_n \mathbf{D}$ at each time step. Note that the time step $\triangle t_n$ is, in general, not a constant, therefore it is impossible to preprocess the system by computing the inversion of this large sparse matrix. Inverting a large matrix at each integration step makes real-time simulation impossible for any problem of reasonable size.

To achieve real-time performance, we approximate the distributed mass with concentrated masses.

4.1 Concentration of Mass

The original mass matrix in (1) is an approximation of the inertia property of the continuum, including the total mass and moment of inertia. However this approximation still treats the mass as if it is distributed. In order to avoid potentially inverting a large sparse matrix at each time step, we approximate the matrix $\mathbf{M}$ by a diagonal matrix, which is obtained by lumping its rows ([36]).

The diagonalization process is equivalent to approximating the mass continuum as concentrated masses at each nodal point of the mesh. By doing this, we basically convert the distributed mass to a particle system. At each integration step, each particle "behaves" independently of the other particles. The force acted on each particle, at each instance of the simulation, consists of external forces, such as gravity, and internal forces exerted by the neighboring particles (nodes that share the same element). Unlike a mass-spring system, this particle system does not have an explicitly defined spring structure. Instead the equivalent internal forces on each particle (a node of the mesh) is modeled using elasticity, approximated by nonlinear FEM.

If we then apply Rayleigh damping $\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K}$, with $\beta = 0$, the matrix $\mathbf{M} + \frac{1}{2}\triangle t_n \mathbf{D}$ becomes a diagonal matrix. This simplifies the nonlinear system of equations (8) into a set of independent *algebraic* equations as following:

$$q^i \ddot{u}^i_{n+1} = f^i_{n+1} - r^i_{n+1} - d^i_{n+1} \tag{10}$$

where q^i is the $i-th$ component of the diagonalized $\mathbf{M}+\frac{1}{2}\triangle t_n \mathbf{D}$; $\ddot{u}^i_{n+1}$, f^i_{n+1}, r^i_{n+1} and d^i_{n+1} are the $i-th$ component of $\mathbf{u}_{n+1}$, $\mathbf{F}_{n+1}$, $\mathbf{R}(\mathbf{u}_{n+1})$ and $\mathbf{D}(\dot{\mathbf{u}}_n + \frac{1}{2}\ddot{\mathbf{u}}_n \triangle t_n)$ respectively. Solving this system of equations requires no matrix inversion.

The diagonalization also makes the enforcement of all types of boundary conditions very simple. For natural boundary conditions, we specify the force and compute u^i_{n+1}. For essential boundary conditions, we simply ignore equation (10) and explicitly set the corresponding displacement and velocity to the given values.

It is worth pointing out that the critical time step for an explicit integration scheme is dictated by the largest stiffness in the material. This is why an explicit integration scheme is appropriate for soft tissues, which are "soft" in all directions (although not necessarily isotropic), while it is not appropriate for cloth simulation [4].

5 Haptic Display

We provide force feedback by simulating the collision between the deformable object and the virtual proxy. To provide stable haptic feedback within limited computational power, we run full FEM simulation at a low frequency. The required high frequency haptic feedback is obtained by interpolating between the simulated states.

5.1 Collision with the Proxy

A virtual proxy is a rigid object with a piecewise differentiable surface. Usually a proxy has a very regular shape, such as a sphere or a cylinder. However In this section, we will discuss collisions using a general proxy.

The popular penalty methods [29, 28, 30] model the collision by adding an artificial spring of large stiffness at the point of collision. This stiff spring requires tiny integration time steps to stably simulate a collision. Various experiments show that the ratio between a collision free integration time step and that of a penalty collision is on the order of hundreds if not more.

This tempts us to develop new collision-handling methods that avoid adding extra artificial stiffness into the system. We will illustrate our collision-handling method, using a special case: collision between a rigid proxy and a single node of the FEM mesh of the deformable body. (figure 4). Later in this section, we will show that it is straightforward to extend this method to handle general collisions between the virtual proxy and the deformable body .

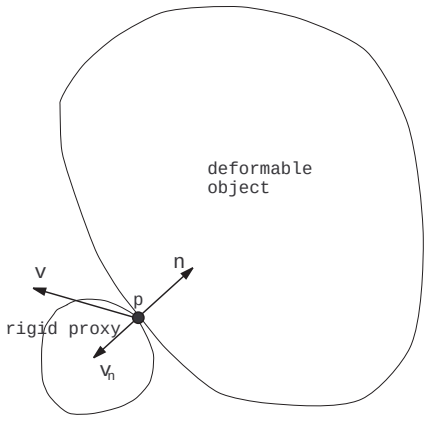

Figure 4: A rigid proxy collides with a soft object.

Consider the collision between a moving deformable body and a moving rigid virtual proxy (figure 4). To simplify the discussion, we use the moving frame attached to the proxy instead of the fixed world frame. Namely all quantities are relative to the moving proxy. Assume that at time t_n, the node p on the deformable object, with *relative* velocity $\hat{\mathbf{v}}(p)_n$, is colliding with the rigid surface of outward normal $\hat{\mathbf{n}}$. The non-penetration constraint requires that the normal component of the relative velocity of point p drops to zero at the moment of collision in the moving frame. Unlike a rigid body collision, the flexible body will maintain contact with the rigid body for a nonzero period of time. We enforce the non-penetration constraint at node p by setting the normal component of $\hat{\mathbf{v}}(p)_{n+1}$ to zero as following:

$$\hat{\mathbf{v}}(p)_{n+1} = \hat{\mathbf{v}}(p)_n + (\hat{\mathbf{v}}(p)_n \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} \tag{11}$$

By equation (9), we get

$$\hat{\mathbf{a}}(p)_{n+1} = \frac{2\hat{\mathbf{v}}(p)_{n+1}}{\triangle t_n} - \frac{2\hat{\mathbf{v}}(p)_n}{\triangle t_n} - \hat{\mathbf{a}}(p)_n \tag{12}$$

If we choose $\triangle t_{n+1} = \triangle t_n$, [2] by equation (7), we have

[2] This does not mean that the entire simulation has to use a constant time step. Indeed the simulation can still use variable time step. This constraint (choice) is only enforced at collision time.

$$\hat{\mathbf{u}}_{n+2} \cdot \hat{\mathbf{n}} = \hat{\mathbf{u}}_n \cdot \hat{\mathbf{n}} \quad (13)$$

This shows that the non-penetration constraint is enforced after two time steps, because there is no relative motion of the deformable body normal to the surface of the rigid proxy.

This collision-handling integration scheme can be considered a special case of impulse [3]. For rigid body collisions, an impulse requires extremely small time steps for numerical integration because the rigid body collision is considered to occur instantaneously. However, for deformable body collisions, the collision time is finite. By delaying the non-penetration constraint by two time steps, we are able to integrate the impulse using large time steps.

By plugging equation (11) into equation (9), we can compute the equivalent acceleration at point p. Then we can use equation (10) to compute the equivalent impulse exerted at point p of the deformable object, which is a force normal to the collision surface. The reaction force exerted on the virtual proxy has the same quantity as this impulse, but in the opposite direction. This equivalent impulse also enables us to compute the Coulomb friction and simulate a frictional collision, and provide friction feedback.

This collision integration scheme can be generalized to a general haptic interface. A general haptic interface involves multiple virtual proxies (for instance, a virtual hand), therefore multiple point contacts. Since the system is decoupled, such a collision is modeled as a set of simultaneous *independent* single point collisions.

Unlike a general impulse [3, 13, 14], we do not have to distinguish the case that the deformable object bounces quickly away from the virtual proxy and that the proxy sticks to or slides on the surface of the deformable object. The bouncing collisions, the sticking contacts and the sliding contacts are handled by exactly the same collision integration scheme, with no extra computational cost.

Our approach is different from that of Ruspini *et al*[25, 23]. Instead of explicitly minimizing the distance between the current configuration and the goal configuration, upon collision, we let physics naturally guide the motion of the virtual proxy. The motion is more physically realistic, compared to that obtained by a purely geometric minimization.

5.2 Haptic Interpolation

While the graphic display of the global deformations requires an update rate of only 30Hz, the stable and smooth haptic display requires an update rate of at least 1000Hz. It is impossible to simulate the global deformation at such a high frequency, with a desktop computer. Note that each graphic frame usually requires multiple integration steps, because the explicit integration scheme has to be smaller than the critical time step to be stable. Therefore although the graphic display is at 30Hz, our system actually simulates the global deformations and the collision between the proxy and the deformable object at a slightly higher frequency[3]. To display haptics at 1000Hz or higher, we will interpolate the haptics between two simulated states using the necessary high frequency.

Given the simulated states at time t_n and t_{n+1}, it is straightforward to interpolate the haptics between them. Basically any interpolation scheme, such as a simple linear interpolation, will do. The problem is that at time t_n, we do not have the information about t_{n+1}. Our solution to this problem is that we simply delay the entire simulation display, both graphically and haptically, by one integration time step. This intentional time lag lasts a few miliseconds. For a virtual interaction with soft objects, such a small lag in time is within the tolerance of human perception. The advantage of such a time delay is that we have already simulated the state at time t_{n+1} when we display the graphics and haptics at time t_n, which makes the haptics interpolation from time t_n to t_{n+1} straightforward.

[3]Each explicit integration step is a full simulation step.

6 Conclusions and Future Work

We presented a haptic simulation system that allows a human operator to interact with 3D global deformations in real time. Due to the distortion associated with linear strain, we simulate the global deformation using geometrically nonlinear finite element methods. The nonlinear FEM formulation is derived from the application of the nonlinear exact strain.

It is in general too expensive to solve such a nonlinear FEM system in real time. In order to achieve real-time performance, we diagonalize the mass matrix *approximately*. This diagonalization is equivalent to converting the distributed mass to a particle system of concentrated mass.

In some sense this approach combines the best aspects of the FEM and mass-spring models. A mass-spring model is inaccurate in its mathematical formulation; however it is cheaper to solve because it is a diagonal system from the very beginning, and it does not introduce any geometric distortion. The FEM model is more accurate in its mathematical formulation of material behavior. But a linear FEM has distortion for large motion and deformation. A diagonalized nonlinear FEM approach models the material behavior with more accuracy than a linear model, and it is still cheap to solve and has no distortion.

Since a stable haptic display requires force computation at a much higher frequency than that required by real-time graphics, we introduced a simple interpolation technique by intentionally delaying the display (both graphic and haptic) by one full simulation cycle. This takes advantage of the fact that human perception tolerates a small delay of a few miliseconds. Such a delay turns a complicated extrapolation into a simple interpolation. We do recognize the possibility of extrapolating haptics without time delay, by estimating a constant local stiffness. However this extrapolation is more computationally expensive than our interpolation technique.

Currently our system is able to produce real-time graphics and haptics for a mesh of several hundred vertices. We are experimenting with the relationship between the stiffness of soft objects and the maximum time delay that can be tolerated by human operators. Our experience suggests that the softer the object is, the longer delay the human operator can tolerate.

7 Acknowledgement

We thank Panayiotis Papadopoulos for sharing with us his FEM expertise and Jonathan Shewchuk for his insight in 3D meshing. We also thank Brett Wilson for his diligent work on interfacing the digital glove.

References

[1] Y. Adachi, T. Kumano, and K. Ogino. Intermediate representation for stiff virtual objects. *Proceedings of IEEE Virtual Reality Annual International Symposium*, pages 203–210, 1995.

[2] David Baraff. Interactive simulation of solid rigid bodies. *IEEE Computer Graphics and Applications*, 15:63–75, 1995.

[3] David Baraff and Andrew Witkin. Dynamic simulation of non-penetrating flexible bodies. In *Computer Graphics: Proceedings of SIGGRAPH*, pages 303–308. ACM, 1992.

[4] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Computer Graphics: Proceedings of SIGGRAPH*, pages 303–308. ACM, 1998.

[5] David Chen. *Pump It Up: Computer Animation of a Biomechanically Basded Model of Muscle Using the Finite Element Method*. PhD thesis, MIT, 1992.

[6] Stéphane Cotin, Hervé Delingette, and Nicholas Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transcation on Visualization and Computer Graphics*, 5(1):62–73, January-March 1999.

[7] M. Finch, V. Chi, R. M. Taylor II, M. Falvo, S. Washburn, and R. Superfine. Surface modification tools in a virtual environment interface to a scanning probe microscope. *Proceedings of 1995 Symposium on Interactive 3D Graphics*, pages 13–18, April 1995.

[8] Sarah F. Gibson and Brian Mirtich. A servey of deformable models in computer graphics. Technical Report TR-97-19, Mitsubishi Electric Research Laboratories, Cambridge, MA, November 1997.

[9] Doug L. James and Dinesh K. Pai. Artdefo: Accurate real time deformable objects. *Computer Graphics: Proceedings of Siggraph*, pages 65–72, August 1999.

[10] E. Keeve, S. Girod, P. Pfeifle, and B. Girod. Anatomy-based facial tissue modeling using the finite element method. *IEEE Visualization*, 1996.

[11] W. R. Mark, S. C. Randolph, M. Finch, J. M. Van Verth, and R. M Taylor II. Adding force feedback to graphics systems: Issues and solutions. *Computer Graphics: Proceeding of Siggraph*, pages 447–452, August 1996.

[12] M. Minsky, M. Ouh-Young, M. Steele, F. P. Jr. Brooks, and M. Behensky. Feeling and seeing: Issues in force display. *Computer Graphics: Proceedings of 1990 Symposium on Interactive 3D Graphics*, pages 235–243, 1990.

[13] Brian Mirtich and John Canny. Impulse-based dynamic simulation. In K. Goldberg, D. Halperin, J.C. Latombe, and R. Wilson, editors, *The Algorithm Foundations of Robotics*. A. K. Peters, Boston, MA, 1995. Proceedings from the workshop held in February, 1994.

[14] Brian Mirtich and John Canny. Impulse-based simulation of rigid bodies. In *Symposium on Interactive 3D Graphics*, New York, 1995. ACM Press.

[15] G. Celniker nad G. Gossard. Deformable curve and surface finite elements for free form shage design. *Computer Graphics*, 25(4), 1991.

[16] M. Ouh-Young. *Force Display in Molecular Docking*. PhD thesis, UNC Chapel Hill, Februry 1990.

[17] S. Peiper, J Rosen, and D. Zeltzer. Interactive graphics for plastic surgery: A task-level analysis and implementation. In *Symposium on Interactive 3D Graphics*, 1992.

[18] E. Promayon, P. Baconnier, and C. Puech. Physically-based deformations constrained in displacements and volume. In *EUROGRAPHICS*, 1996.

[19] X. Provot. Deformation constrains in a mass-spring model to describe rigid cloth behavior. *Computer Interface*, 1995.

[20] J. N. Reddy. *An Introduction to the Finite Element Method*. McGraw-Hill, Inc., 2nd edition, 1993.

[21] Diego Ruspini. Adding motion to constraint based haptic rendering systems: Issues and solutions. *Proceedings of the Second PHANToM User's Group Workshop*, October 1997.

[22] Diego Ruspini and Oussama Khatib. Dynamic models for haptic rendering systems. *Advances in Robot Kinematics*, pages 523–532, June 1998.

[23] Diego C. Ruspini, Karsimir Kolarov, and Oussama Khatib. The haptic display of complex graphical environments. *Computer Graphics Proceedings*, pages 345–352, August 1997.

[24] Diego C. Ruspini, Krasimir Kolarov, and Oussama Khatib. Rubust haptic display of graphical environments. *Proceedings of The First PHANToM User's Group Workshop*, September 1996.

[25] Diego C. Ruspini, Krasimir Kolarov, and Oussama Khatib. Haptic interaction in virtual environments. *The Proceedings of the International Conference on Intelligent Robots and Systems*, September 1997.

[26] Kenneth Salisbury. An overview of haptics research at mit's ai lab. *Proceedings of The First PHANToM User's Group Workshop*, September 1996.

[27] M. A. Srinivasan and J. K. Salisbury. Chapter 4: Haptic interfaces. *Virtual Reality: Scientific Techonological Challenges*, 1994.

[28] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. *Computer Graphics*, 22, August 1988.

[29] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. *Computer Graphics*, 21, July 1987.

[30] D. Terzopoulos and K. Waters. Physically-based facial modeling, analysis and animation. *Journal of Visualization and Computer Animation*, 1990.

[31] Sundar Vedula and David Baraff. Force feedback in interactive dynamic simulation. *Proceedings of The First PHANToM User's Group Workshop*, September 1996.

[32] K. Waters. A muscle model for animating three-dimensional facial expression. *Computer Graphics*, 21(4), July 1987.

[33] H. M. Westergaard. *Theory of Elasticity and Plasticity*. Dover Publications, Inc., 1964.

[34] A. Witkin and *et al*. An introduction to physically based modeling. Course Notes, 1993.

[35] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method: Basic Formulation and Linear Problems*, volume 1. McGraw-Hill Book Company, 4th edition, 1989. linear finite element method, linear elasticity.

[36] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method: Solid and Fluid Mechanics Dynamics and Non-Linearity*, volume 2. McGraw-Hill Book Company, 4th edition, 1989.

[37] C. B. Zilles and J. K. Salisbury. A constraint-based god-object method for haptic display. *ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 1994, Dynamic Systems and Control*, 1:146–150, November 1994.