DYNAMIC MODELS FOR HAPTIC RENDERING SYSTEMS

DIEGO RUSPINI AND OUSSAMA KHATIB
Robotics Laboratory
Department of Computer Science,
Stanford University, Stanford, CA 94305-9010
email: ruspini,khatib@cs.stanford.edu

Abstract: To create a compelling interactive virtual experience, haptic systems must incorporate methods to allow for the specification of and interaction with complex dynamic environments. In this paper we discuss some of the issues encountered and the solutions developed for incorporating dynamic motion in our haptic rendering library "HL." The objective in the development of this library is to provide haptic capabilities to users who may have little or no familiarity with robotics, haptics or control.

1. Introduction

For successful development of real world applications, haptic rendering systems must provide the ability to manipulate, interact, and modify a virtual environment. While movements displayed at 30 frames a second appear visually continuous, the same movements would feel rough and uneven when updated at that same rate on a haptic system. To achieve realistic smooth motion, the haptic rendering system must incorporate methods to allow motion and force interaction to be specified and executed at a very high update rates, e.g. greater 1000Hz.

This paper discusses some of the techniques we have developed to incorporate dynamics into the "HL" constraint based haptic rendering library (Ruspini, 1997). The goal of this library is to make haptic rendering capabilities available to a graphic applications developer, who may have little or no knowledge of haptics or robotics. As such, the library is designed to take as input the specification of arbitrary graphical models and display them haptically without the need for any extensive translation. To accomplish this the library emulates as much possible the syntax and conventions used by common graphic systems. The library is build on the foundation of using a representative virtual object, a "proxy" discussed in section 2, to model all interaction between the user and the virtual environment. This concept is used not just to model surface contact but in addition friction, shading and texture. In this paper, the virtual proxy concept is extended to the incorporation of motion in the haptic environment.

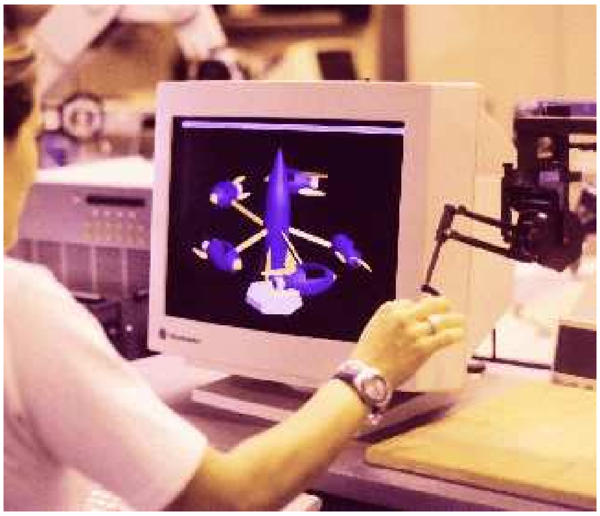

Figure 1. User haptically interacting with a dynamic virtual environment

In particular we will look at two common model types. When modeling mechanical systems composed largely of rigid bodies, a articulated linkage can be constructed and used to model the dynamic behavior of the bodies. Models composed of deformable or soft non-rigid bodies, however, cannot be easily described with a single efficient general purpose model. In these cases an application program, with domain specific knowledge, can use the haptics library to update the virtual objects to adjust the surface based on the forces applied by the user. Given the often limited bandwidth between the user application and the haptic controller it is usually not possible to update the model at a rate sufficient to give the feeling of a continuous free-form deformation. In this paper we propose a method to "morph" between the discrete and infrequent surface definitions and restores the apparent continuous nature of the surface's motion.

2. Constraint Based Rendering in Static Environments

Earlier haptic rendering methods model surface contacts by generating a repulsive force proportional to the amount of penetration into an obstacle. These penalty based methods behaved poorly when the environment contained thin or overlapping obstacles. Constraint based methods, first introduced by Zilles et. al (Zilles, 1994) define a constrained point, or representative object that is prevented from penetrating the surfaces in the environment. In our approach, a finite sized, massless "proxy" substitutes in the virtual environment for the physical finger or probe. The "virtual proxy" can be viewed as if connected to the user's real finger by a stiff spring. As the user moves his/her finger in the workspace of the haptic device he/she may pass into or through one or more of the virtual obstacles. The proxy, however, is stopped by the obstacles and quickly moves to a position that minimizes its distance to the user's finger position subject to the constraints in the environment. A haptic device, a small lightweight robotic manipulator (Figure 1), is used to generate the forces of the virtual spring which appear to the user as the constraint forces caused by contact with a real environment. An example of this motion is shown in Figure 2.

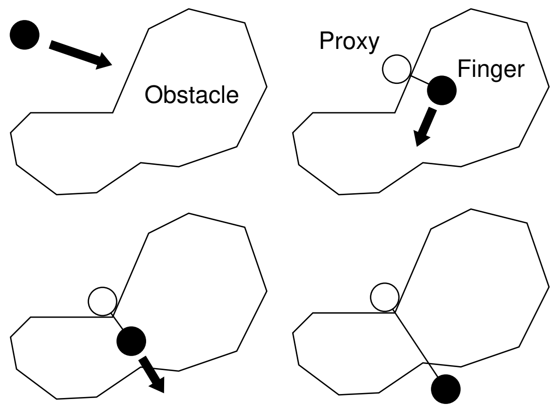

Figure 2. The virtual proxy moves to locally minimize the distance to the user's finger position subject to the constraints in the environment

From an algorithmic point of view it can be easily seen that the motion of the proxy is very similar to that of a robot reactively moving towards a goal (the user's finger) under the influence of an artificial potential field (Khatib, 1986). When unobstructed, the proxy moves directly towards the goal. If the proxy encounters an obstacle, direct motion is not possible, but the proxy may still be able to reduce the distance to the goal by moving along one or more of the constraint surfaces. When the proxy is unable to further decrease its distance to the goal, it stops at the local minimum configuration. This analogy also holds for the goal (the user's finger) which through forces applied by the haptic device is pushed towards the proxy's (the robot's) position.

The advantages of using a constrained object over a penalty based approach is that the position and movement of the proxy can be completely specified and controlled to ensure that it will not pass through or into the obstacles in the environment. Constraint based methods, therefore, permit haptic rendering with infinitely thin obstacles, such as obstacles defined only by their boundary surface representation. The ability to model these environments is crucial for building a library for general graphics applications, where surface representations are widely used. This concept has been also used to render a number other haptic attributes including, shading of curved objects (force shading), friction (static, dynamic, viscous), texture, and stiffness (Ruspini, 1997).

3. Adding Motion to Virtual Environments

Incorporating motion in the virtual environment substantially increases the complexity of haptic rendering over that of static scenes. As illustrated in figure 3(a), motion in the environment can be treated by considering an additional dimension of time added to the configuration space associated with the proxy. The goal of the proxy control loop is to move the proxy to the configuration with minimal distance with respect to the user's finger in the configuration-time space of the current servo clock cycle. When rotations are permitted in the virtual environment, it can be easily shown (see Figure 3(a)) that the resulting space-time obstacle may be quite complex and non-convex even if the original obstacle is composed only of simple convex pieces.

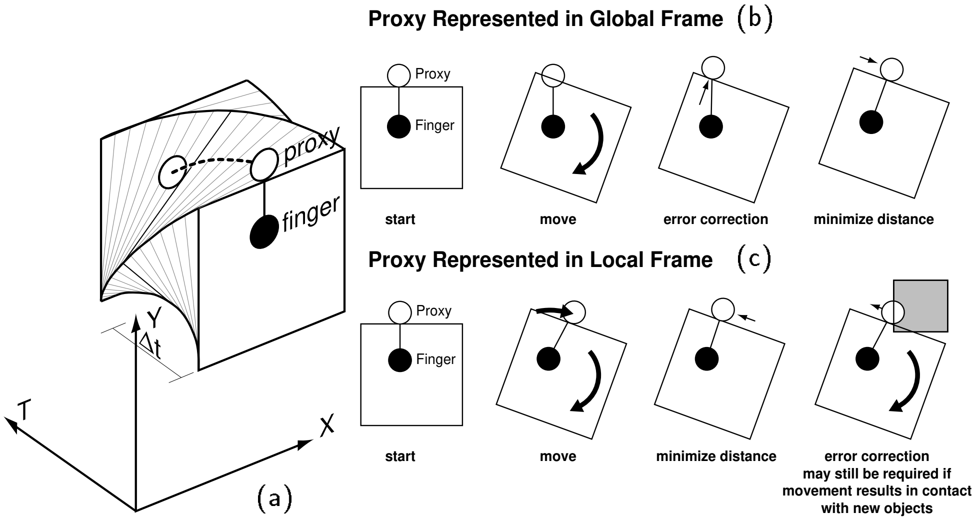

Figure 3. When rotation is permitted simple objects can result complex space-time obstacles(a). An iterative method is used to move the proxy to its minimal energy configuration(b). When a static contact constraint exists the proxy's position is more effectively represented in the local frame of the obstacle(c).

3.1. ITERATIVE UPDATE SOLUTION

Since no general closed form solution exists for finding the minimal configuration in this complex space, we propose an iterative method with a fast convergence time. Because of the proxy's finite size it is possible to allow it to penetrate the obstacle to certain extent before it passes though a surface and appears on the opposite side. In our implementation the proxy is modeled as a sphere of radius r_{proxy}. If the maximum penetration distance is guaranteed to be less than r_{proxy} then the constraint planes (Ruspini, 1997) can be used to eliminate the detected error. The maximum velocity, therefore, of a single surface, for which the proxy can be guaranteed not to penetrate the environment, is $v_{\max} = r_{\text{proxy}}/\Delta t$, where Δt is the amount of time required to complete one iteration.

In our implementation, the iteration is divided into three stages. In the first stage, the objects in the environment are moved over some time period Δt. In the next stage the constraint planes are found as in (Ruspini, 1997) and used to attempt to move the proxy to nearest free space configuration. In the last stage, the same distance minimization is applied to move the proxy to a closer (minimal energy) configuration. This iteration is illustrated in Figure 3b.

Note that stages two and three may need to be repeated for multiple iterations because the proxy may continue to encounter new obstacles as it converges towards a minimal configuration. Additionally since the movement of the obstacles is made without regard to the constraints imposed by the rigidity of the proxy, the obstacles may move in such a way that no possible collision-free configuration of the proxy can be found. Such a situation can occur if, for example, two large parallel slabs were to come together, effectively eliminating the local free-space around the proxy. While these situations can be resolved using constraint based techniques like those discussed in (Baraff, 1994; Ruspini, 1997), these systems currently remain too slow for the real-time requirements of haptic rendering. In practice these situations rarely appear, and are treated by allowing the proxy to pop-through one of the obstacles.

3.2. ITERATION WITH STATIC CONTACT

A special case needs to be considered when the proxy is in static contact with one of the obstacles in the environment. When such a situation occurs the movement of the object and the subsequent error correction can cause the proxy to drift from its original contact point. Such a situation appears disconcerting to an operator holding or moving an object. To remedy this situation, when the proxy is in static contact with an object, its position will be stored in the local frame of the obstacle. When the object is moved the proxy is also moved in the same frame and maintains the same contact point. Although a new collision with the statically constrained obstacle can not occur, the error correction and minimization stages should still be performed. This ensures that the static friction constraint is not violated by a collision with another obstacle, and establishes that the proxy is still in static contact with the original object.

4. Modeling Interaction with Tree Linkages

Our previous discussion has been limited to the kinematic aspects of the motion of the virtual environment. To create an engaging virtual world, the user must be able to manipulate and dynamically interact with the virtual objects. Inspired by the real world, physically based models are the most intuitive representations for modeling the behavior of the virtual environment. When modeling a mechanical system or articulated figure, it is often convenient to describe the system as a multi-chained linkage of rigid bodies connected by simple one degree of freedom (DOF) joints. Null links (links containing no mass/inertia) can be used to allow more complex joints to be constructed by specifying several simple joints in series. The position of each joint (rotational, prismatic, screw) can be specified by a joint coordinate $\Theta_i, 1 \leq i \leq n$, where n is the number of DOF of the system. The configuration of the entire system can then be specified by the joint space vector $\Theta = [\Theta_1 \ldots \Theta_n]^T$. Some example tree linkages are shown in figure 5. The forward dynamics equations of motion of such a system can be used to obtained the joint space accelerations of the system. These equations can be written as:

$$\ddot{\Theta} = M(\Theta)^{-1}(\Gamma - b(\Theta, \dot{\Theta}) - g(\Theta)), \tag{1}$$

where $M(\Theta)$ is the positive definite kinetic energy matrix, $b(\Theta, \dot{\Theta})$ the centrifugal coriolis vector, $g(\Theta)$ the gravity force vector and Γ is the vector representing the internal and external torques applied to the system either through internal actuation or external forces applied by the environment. A simplified set of equations that neglects the centrifugal coriolis and some dynamic coupling terms is used in our current system. We are in the process of converting the system to make use of an extremely fast variation of Featherstone's articulated body solution (Featherstone, 1987) to permit the simulation of very complex dynamic models at real-time rates (Chang, 1998).

When a collision occurs, between the proxy and an object(s) in the environment, a force is applied to the user simulating a contact with the surface. In a dynamic environment an equal and opposite force F_c is applied at the contact point(s) which may induce accelerations on the virtual system. The corresponding joint torque vector is given by

$$\Gamma_{ext} = J_c^T f, \tag{2}$$

where J_c is the Jacobian of the contact point c, such that the velocity v of the contact point is given by $v = J_c\dot{\Theta}$. For a simple PD controller the force applied to the environment can be given by

$$F_c = k_p(p_{\text{proxy}} - p_{\text{finger}}), \tag{3}$$

where k_p is the positional gain, and $p_{\text{proxy}}, p_{\text{finger}}$ are the current positions of the proxy and finger respectively. Note that this force is in general not sufficient to prevent penetration between the proxy and objects in the environment as this equation does not incorporate the internal constraints of the proxy or other objects. A more complete solution to computing the contact forces for rigid body simulation can be found in (Ruspini, 1997). This simplified model, however, is sufficient for simulating the interactions found in most haptic environments. Once the joint space accelerations are known the equations of motion for the system can be integrated, from a given initial joint space configuration and velocity, to obtain the motion for the entire system over time.

While the joint space configuration vector Θ describes the relative motion between the rigid bodies in the environment; for many interesting mechanical systems this vector may not be the most appropriate parameterization, nor may it represent the true number of degrees of freedom of the system. More complex, closed chain, geared, or otherwise constrained mechanical systems can be constructed by introducing an additional level of abstraction. Instead of defining each link as a independent degree of freedom, each link variable is specified as the function of one or more configuration/animation variables *(avars)*. In the general case

$$\Theta = f(q) \tag{4}$$

for some set of *avars* $q = [q_1 \ldots q_m]^T, 1 \leq m \leq n$. Differentiating twice we obtain the relationship between the accelerations of the *avars* and the joint space configuration.

$$\dot{\Theta} = J\dot{q}. \tag{5}$$

$$\ddot{\Theta} = J\ddot{q} + \dot{J}\dot{q}. \tag{6}$$

where J is the configuration space Jacobian given by

$$J = \begin{bmatrix} \frac{\partial f_{\Theta_1}}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \cdots & \frac{\partial f_1}{\partial q_m} \\ \frac{\partial f_2}{\partial q_1} & \frac{\partial f_2}{\partial q_2} & \cdots & \frac{\partial f_2}{\partial q_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial q_1} & \frac{\partial f_n}{\partial q_2} & \cdots & \frac{\partial f_n}{\partial q_m} \end{bmatrix}. \tag{7}$$

Combining equation 1 and 6 we can obtain the equations of motion in the new configuration space q

$$\ddot{q} = J^+ \left[M(\Theta)^{-1}(\Gamma - b(\Theta, \dot{\Theta}) - g(\Theta)) - \dot{J}\dot{q} \right] \tag{8}$$

where $J^+ = (J^T J)^{-1} J^T$ is the left inverse for the over-constrained system mapping $\Theta \rightarrow q$. Given an initial configuration space position and velocity the forward dynamic equations of motion of the system can be integrated and the joint angles of linkage found using equation 4. An example of this mapping can be seen in Figure 5(c) which displays an interactive roller coaster. While each car of the coaster moves in a six dimensional space ($x,y,z,roll,pitch,yaw$), resulting in a joint space configuration vector of size $6 \times 3 = 18$, the cars are constrained to lie on the track (a one dimensional curve in the 18 dimensional joint space of the system) resulting in a system with only one true degree of freedom.

5. Proxy Blending

Some models can not be conveniently decomposed into an articulated linkage. Objects that are composed of a malleable material like clay, or soft tissue have many more DOF than can be efficiently represented by a mechanical linkage. In these cases the object model will need to be updated by the application program which can exploit specific knowledge about the material being modeled. In many cases it is not possible for the application program to update the object model at a rate sufficient for haptic rendering ($>$ 1000Hz).

To bridge this gap between the update rates of the simulation and the haptic rendering processes we can exploit an interesting asymmetry in the bandwidth of human tactile interaction. While humans can sense force vibrations in excess of 500Hz, the maximum human motion output bandwidth is only $\sim$10Hz, and typically much less, on the order of 1 or 2 Hz. In many simulations only the low-frequency motions are of interest for display. In these situations, by smoothly transitioning between discrete low-frequency model updates we can give the user a sensation of a continuous free-form deformation. The underlying surface model can therefore be updated at the same rate as it is displayed on the computer display (10-30Hz).

To haptically "morph" between the old and new object definitions two proxies are momentarily created. One proxy is constrained only by the old environment, while the other is constrained only by the new definition. Over a blending period, both the old and new proxies are updated according to the movements of the user's finger. During this period, the goal point of the user's finger is smoothly interpolated from the old proxy's to the new proxy's location. When the blending period expires the old proxy and object definition are deleted and the rendering process continues on the updated environment. This process is illustrated in Figure 4. Preliminary results indicate that the "blending" is sufficient to restore the apparent continuous nature of the motion seen on the visual display. This technique can also be used to ensure a gradual change in the output force when objects are first created and appear or change instantaneously in the virtual environment.

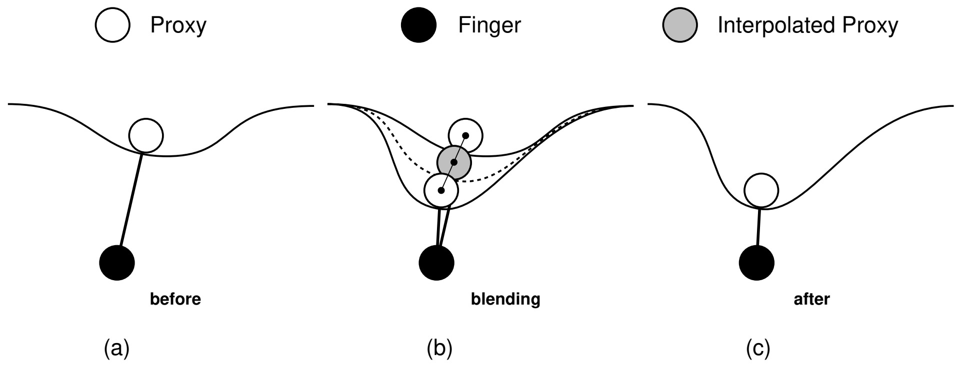

Figure 4. Proxy Blending: During a (b) blend the actual proxy is interpolated from a proxy constrained by the object definition before (a) and after (c) the blending period.

6. Results

The techniques we have described were used to model a variety of virtual environments. At present, several restrictions have been placed on the complexity of the modeled environment. These ensure the real-time update rate of the haptic update loop and simplify the specification of the dynamic models. Joint angles are currently limited to be the function of at most one *avar* value. That is $\Theta_i = f_i(q_j)$ for a given joint angle $\Theta_i, 1 \leq i \leq n$ and $q_j, 1 \leq j \leq m$. In addition we currently restrict the function $f_i()$ to be a piecewise linear function of q. This restriction insures that $\dot{J} = 0$ and that J^+ can be computed in linear time. Figure 5 demonstrates some of the virtual environments that can be haptically displayed by our system.

7. Conclusion

The virtual proxy concept developed for the haptic display of static environments has been successfully extended to dynamic and moving virtual worlds. An iterative method for updating the constrained proxy's position given the movement of the obstacles was presented that converges quickly and reliably for real-time haptic applications. Movement can be obtained from the equations of motion of an articulated linkage or in the case of non-rigid objects from the graphics application even at a slow rate and blended to give the sensation of a smoothly changing deformable body. In the future we hope to continue to extend and generalize the types of virtual environments that can be specified and modeled by our library and help extend the range of applications that can benefit from the use of haptic technology.

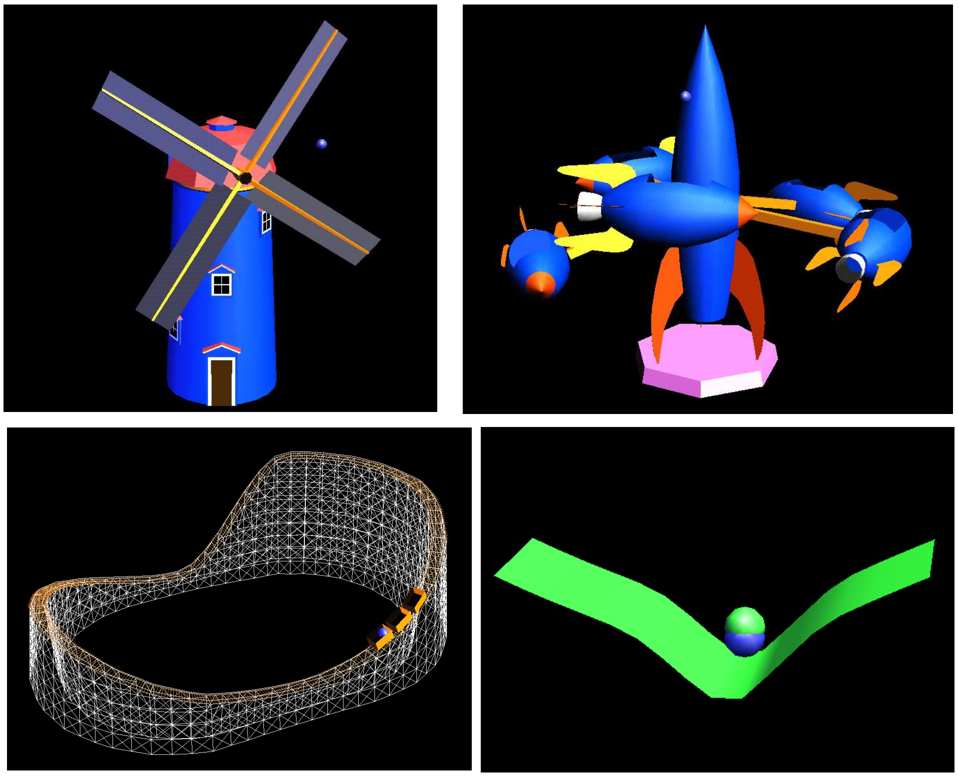

Figure 5. Windmill, Rocket Carousel, a Haptic Roller Coaster, and Interactively Deformable Membrane are a few of the models that can be rendered by our system

References

D. Baraff, "Fast Contact Force Computation for Nonpenetrating Rigid Bodies," *SIGGRAPH 94 Proceedings*, (August 1994), pp. 23-34.

K. Chang, "Efficient Dynamic Control and Simulation of Robotic Mechanisms," internal report.

J. Craig, "Introduction to Robotics Mechanics and Control," *Addison-Wesley*, 1989.

R. Featherstone. *Robot Dynamics Algorithms.* Kluwer, 1987.

H. Goldstein, "Classical Mechanics," Addison-Wesley Publishing Company Inc., 1980.

T. Kane, *Dynamics: Theory and Applications*, McGraw-Hill, 1985.

O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robot," *International Journal of Robotics Research*, 5(1), pp. 90-98.

D. Ruspini, K. Kolarov and O. Khatib, "The Haptic Display of Complex Graphical Environments." SIGGRAPH 97 Proceedings, (August 1997), pp. 345-352.

D., Ruspini, and O. Khatib, "Collision/Contact Models for the Dynamic Simulation of Complex Environments," *Workshop on the Dynamic Simulation, IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, IROS '97, Genoble, France, 1997.

C. Zilles, J. Salisbury, "Constraint-based God-object Method for Haptic Display." ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 1994, *Dynamic Systems and Control 1994*, vol. 1, pp. 146-150.