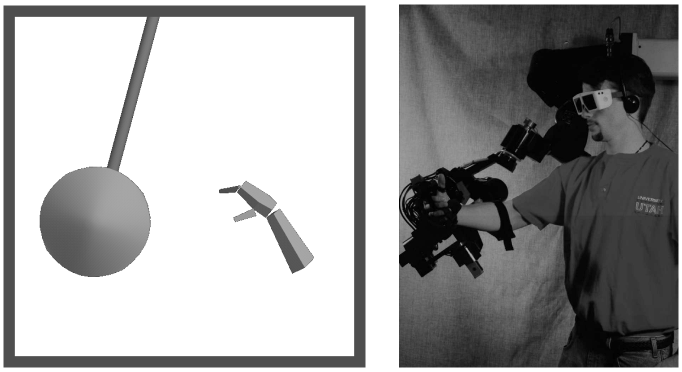
Proceedings of Symposium on Haptic Interfaces
ASME International ME'97 Congress and Exposition
November 15-21, 1997, Dallas, Texas

MANEUVERABLE NURBS MODELS WITHIN A HAPTIC VIRTUAL ENVIRONMENT

Thomas V. Thompson II Donald D. Nelson Elaine Cohen John Hollerbach
University of Utah
Computer Science
50 S Central Campus Dr Rm 3190
Salt Lake City, UT 84112-9205
Email: {tthompso,dnelson,jmh,cohen}@cs.utah.edu

ABSTRACT

Haptic rendering should not be limited to immobile polygonal models. This paper presents a system that performs Direct Parametric Tracing (DPT) on maneuverable Non-Uniform Rational B-Splines (NURBS) models without the use of an intermediate representation. The system distributes computation between an advanced CAD modeling system and a microprocessor controller for a Sarcos force-reflecting exo-skeleton arm. Methods ranging from model manipulation to simulation of the dynamics of simple assemblies are discussed. Solutions to synchronicity issues that arise in such a distributed system are also described.



(a) (b)

Figure 1. (a) A dynamic pendulum within a haptic virtual environment. (b) The Sarcos Dextrous Arm Master.

1 INTRODUCTION

Current CAD systems offer an array of visual information to aid the designer. Isoline drawings, shaded images and even animations help the designer by providing more information than simple 2D drawings can convey. As CAD models grow more complex, increasingly sophisticated methods are needed to convey the meaning of a design. In fact, with increased model complexity, the relationships between a model's parts often become more important than just the part shape. For this reason, touching and tracing a static model can only add so much to the modeling experience. If the model is an assembled linkage, the designer should be able to put the assembly into motion in order to interactively test its function. In conjunction with visual feedback, haptic rendering and dynamic simulation can, by creating a new channel of feedback, increase the understanding of complex models and add a sense of realism to interactive systems (Hollerbach et al., 1996).

In this paper, we introduce direct haptic rendering of maneuverable sculptured models (Fig. 1a) by extending the direct haptic rendering techniques developed in (Thompson et al., 1997). Furthermore, we have made improvements in smooth transitioning between surfaces and wall model computations, and we are developing a dynamics package. A few dynamic test cases have been constructed to demonstrate the movement of models and to allow this movement to be physically based. These algorithms are tested on a complete system (Fig. 1b) that integrates *Alpha_1*, a research modeling package (Riesenfeld, 1993; Riesenfeld, 1989), with a Sarcos Dextrous Arm Master (Jacobsen et al., 1990).

2 BACKGROUND

The goal of a haptic rendering system is to generate forces that can be applied to a user's hand or arm that will accurately produce a sense of contact with a virtual model. These forces, called restoring forces, prevent penetration into the virtual model and are calculated using a wall model.

Two basic response models can be used: compliance and stiffness.

The compliance model (Yoshikawa, et. al, 1995) approach takes force measurements and uses a control strategy to render acceleration or another form of motion to the user and the virtual object. In the stiffness model approach, position is measured and force is displayed. Wall models based on the stiffness model often have a restoring force proportional (Colgate, 1994) to the penetration depth and in the direction of the surface normal. The stiffness model is the prevalent one for for haptic rendering.

Several types of objects have been haptically rendered. Polygonal models were traced by using techniques illustrated in (Salisbury et al., 1995; Zilles and Salisbury, 1995). Their method uses a system of constraints to track a point on the polyhedrons surface and calculates penetration depth and surface normal from the tracked surface point. They recommend interpolating the surface normals (much like Phong shading in graphics) as a method to portray sculptured models. These systems are often limited to simple models since complex models require a very large number of polygons.

Others (Adachi et al., 1995; Mark et al., 1996) recommend the use of intermediate representations to aid in haptic rendering of complex shapes. These system haptically render the model by using relatively slowly changing planar approximations to the virtual model. This method allows more complex models to be rendered but is limited when trying to approximate surfaces with high curvature.

Free-form surfaces have been traced using distribution functions (Adachi, 1993) which allows for quality tracing of smooth surfaces. Constructing models using distribution functions is difficult and it is often necessary to use complex numerically unstable high order functions.

Parametric surfaces have become the surface representation of choice in CAD/CAM. Parametric surfaces such as NURBS have the advantage of a compact representation, higher order continuity, and exact computation of surface normals which are all useful in complex, realistic virtual environments (Snyder, 1995). Direct haptic rendering of NURBS models was demonstrated using Direct Parametric Tracing (Thompson et al., 1997). Using this method, designers can touch and trace a CAD/CAM model at interactive rates without the use of an intermediate representation.

While previous papers discuss haptically rendering various types of models, they do not discuss the issues involved in allowing the virtual models to move within the environment. The method must allow the models to be traced at interactive rates while at the same time maintain consistency between the visual and haptic versions of the model. In addition, problems regarding network latency and display rates must be solved for virtual environments with movable models to be realized.

3 SYSTEM OVERVIEW

Several constraints influence the design of a haptics system (Hollerbach et al., 1997). Among the requirements for a design environment that includes haptics are an interactive visual display, low latency haptic processes, real-time computation, and careful memory usage. While rates of twenty frames per second are often considered acceptable in VR systems, cycle rates of several hundred Hz are necessary for a haptics system to maintain stiff virtual surfaces (Minsky et al., 1990).

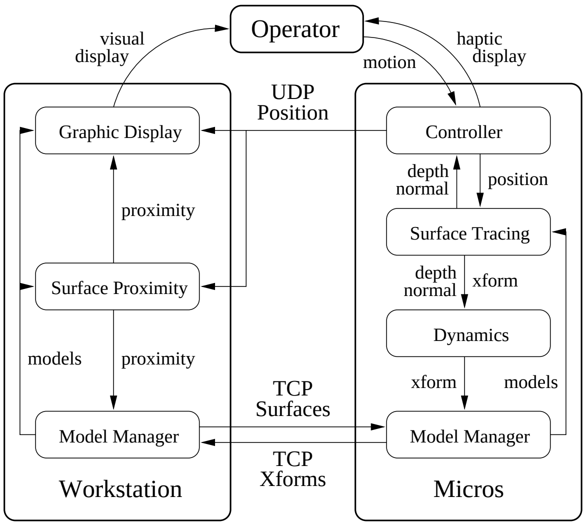


Figure 2. The system is divided into three portions: operator, micros, and workstation.

Our system is distributed to allow the haptics controller to run on low-latency, real-time microcomputer boards, while the graphics workstation generates the visual display and contains the modeling environment (Fig. 2). We refer to the computation on the workstation, typically global in scope, as the *simulation process*, and the computation on the micros, typically local, as the *haptic process*.

We have developed a distributed computation model to maintain model consistency and control the movement of data between the simulation and haptic processes. Global computations occur in the simulation process. When the simulation process detects the potential for an interesting event in the near future, such as contact with a surface, it signals the haptic process, which continues the computation with low latency, local methods. Data caching within the haptic process reduces communication overhead. The haptic process has priority over the simulation process in maintaining model integrity, since final determination of such state changing events as surface contact occur there. Periodic updates of the simulation process keep the data consistent.

We use three communication channels for the various forms of communication in our system (Fig. 2). UDP is used to transmit the current arm position to the simulation process. An unguaranteed protocol can be used since the data is sent continuously and only the most recent position is required. TCP, however, is used when the data must be guaranteed to arrive. Therefore, TCP is used to transmit surfaces and activation records to the haptic process as well as to send transformation matrices to update the models within the simulation process.

4 SIMULATION PROCESS

The simulation process runs within the *Alpha_1* CAD modeling environment and has three main tasks in addition to serving as the design environment: graphical display, managing models, and surface proximity testing.

4.1 Model Manager

The distributed nature of our system forces us to store models on both sides of the system; therefore, it is important that these models be kept consistent or the visual display and the haptic display will not correspond.

As a model is loaded into the environment a copy is sent to the haptic process. The position of these models is determined by a transformation matrix (xform) from model space to world space. Both processes initialize the xform to be equal to the identity matrix.

The state of an individual surface within the haptic process is controlled by the simulation model manager. If the end-effector is close enough to a surface to potentially touch it, then it is deemed proximal. Once this determination occurs, a small activation record consisting of a starting parametric value for DPT and the surface's id is sent to the haptic process. A surface stays *active* in the haptic process until it is no longer deemed proximal. At that time a deactivation record consisting of the surface id is sent to the haptic process and tracking of that surface is discontinued.

4.2 Surface Proximity Testing

Since the computational power of the haptic process is limited, we control the number of surfaces being tracked at any time. Taking into account network latency, the speed at which the user can move the arm, and the servo rate of DPT, we have determined that a distance of $10cm$ is sufficient to be considered proximal. A $5cm$ buffer prevents thrashing between active and inactive determination.

Every surface in the simulation process is checked for proximity. A majority of surfaces are eliminated by a quick bounding box check, with remaining surfaces checked using an algorithm based on a time-critical method that spends less time on objects outside the region of interest (Johnson and Cohen, 1997). Using this approach, we minimize the time spent performing global closest point calculations.

4.3 Graphical Display

An important part of any design system is the graphical display. To increase the level of immersion, a virtual hand that mirrors the movement of the user's hand is displayed (Fig. 3). The virtual hand illustrates the same degrees of freedom present in the Sarcos hand by displaying the forearm, palm, thumb and finger. Actual joint coordinates are used to determine the virtual hands position and orientation within the display. This data is received from the haptic process as a continuous stream of UDP packets. Only the most recent data is used to minimize lag.

Figure 3. Goblet being traced by virtual hand.

The models are located within the display by their associated xform, with the xform value controlled by the model manager within the haptic process. As updates to a model's xform are received the display is updated to maintain consistency between where a user feels an object and where it appears. The combination of the low lag virtual arm and the synchronized visual and haptic display of the models creates a realistic virtual environment.

5 HAPTIC PROCESS

The haptic process runs on a hybrid PowerPC 604 and Motorola 68040 VME system and is a real-time process. The 68040 processor runs the low level torque control servo loop at 1500Hz and does gravity compensation. In parallel, the 604 processor does surface tracing, dynamics and force response calculations at 1500Hz. Networking is also performed on the 604 but at a reduced rate of 50Hz. Only those computations that must be performed for each cycle of the force loop are done in the haptic process.

5.1 Model Manager

The haptic process model manager is responsible for the cache of models. Each model within the haptic process has an associated xform and a list of active and inactive surfaces. Only those surfaces spatially local to the end-effector need be tracked and are designated as active.

Also, this model manager is responsible for supplying surfaces and xforms to the surface tracer. If tracing results in the dynamics package indicating that a model has been moved, then the xform must be updated within the cache. The updated xform is packed into a small packet and sent to the simulation model manager via TCP so that the models remain consistent.

5.2 Surface Tracing

Each active surface is a candidate for contact by the user. For this reason, a local closest point is tracked on each of the active surfaces using DPT. We then apply a mapping to the end-effector to permit the surfaces to move. The collection of local closest points aids in the speed and smoothness of the transitioning algorithm.

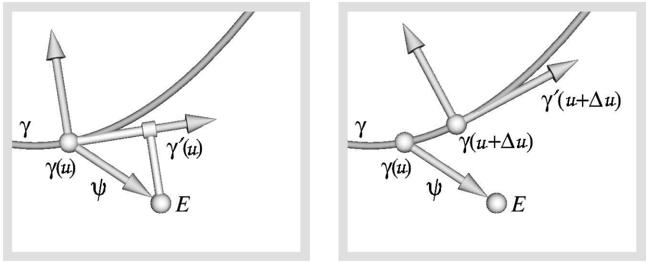
5.2.1 Tracking Phase. We use the tracking algorithm presented in (Thompson et al., 1997). The algorithm has been shown to run at interactive rates and thus is suitable for direct haptic rendering. This method is initialized by the parametric value given as part of the activation record.

We present the DPT method on a NURBS curve, $\gamma(u)$, of order k, defined by the control polygon $\{P_i\}$ and the knot vector $\{u_i\}$. The method uses the previous point on the curve $\gamma(u)$, the tangent vector at $\gamma(u)$, $\gamma'(u)$, and the new end-effector location, E, to determine a new approximate closest point on the curve (Fig. 4).

The parametric velocity of the curve, $\gamma'(u)$, relates changes in position along the curve in Euclidean space to changes in position in parametric space (Eq. 1).

$$\gamma'(u) = \frac{d\gamma}{du} \approx \frac{\Delta\gamma}{\Delta u}. \qquad (1)$$

An approximation for $\Delta\gamma$ is the projection of the vector, ψ, onto the curve tangent at $\gamma(u)$ (Fig. 4a). The curve parametric velocity over the range of movement is found using a first order approximation, the value $\gamma'(u)$.



(a) (b)

Figure 4. (a) Projection of position onto surface tangent plane. (b) New surface point and tangent plane found via parametric projection.

The evaluation point $\gamma(u^*)$, is calculated by refinement (Cohen, 1980). This results in a new knot vector $\{\hat{u}\}$ and a new control polygon $\{\hat{P}\}$ where $\hat{P}_{i^*} = \gamma(u^*)$. The properties of the refined curve result in a greatly simplified form for $\gamma'(u)$ (Eq. 2).

$$\gamma'(u^*) = \frac{(k-1)}{\hat{u}_{i^*+k} - \hat{u}_{i^*+1}}(\hat{P}_{i^*+1} - \hat{P}_{i^*}). \qquad (2)$$

We can use this information to show that Δu can be calculated efficiently by using only the values of the control polygon, the curve knot vector, and the curve order.

$$\Delta u \approx \frac{\langle\, \psi\, ,\, (\hat{P}_{i^*+1} - \hat{P}_{i^*})\, \rangle}{\|\hat{P}_{i^*+1} - \hat{P}_{i^*}\|^2} \left(\frac{\hat{u}_{i^*+k} - \hat{u}_{i^*+1}}{k-1}\right). \qquad (3)$$

This method is essentially the same for surfaces. The projection step requires projection onto the tangent plane of the surface and barycentric coordinates are used to derive Δu and Δv.

5.2.2 Moving Surfaces. The DPT method is designed for end-effector movement and static surfaces. In our system both the end-effector and the surface can be moving. Basically, three approaches can solve this problem. First, the models can be physically transformed as they are moved. There are several drawbacks to this approach. Incrementally transforming objects has been shown to induce numerical errors over time. In addition, the transform calculations are all done when the processing power of the micros is most needed—when the dynamics and tracing are both being calculated.

The other approaches involve storing a transformation matrix for each model. The second approach is to use a xform to transform the active surfaces of each model and perform DPT on them as usual. This approach has a similar drawback as the first method, since all active surfaces must be transformed in order for the tracking to succeed. In fact, transformations must be calculated even when the user is not in contact with a model so that the tracking can still be successful. Fewer surfaces are being transformed, but they may be transformed more often.

We have adopted the third approach which involves using DPT on the original (non-transformed) surface with an end-effector position that has been transformed into model space (Fig. 5).

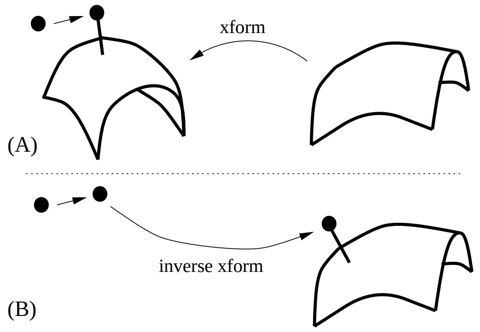


Figure 5. Surface movement transformed into end-effector movement. (A) Surface and end-effector movement. (B) End-effector movement with inverse xform component.

For each model with active surfaces, we transform the end-effector through the inverse of the model's xform. The result is to transform the movement of the model into a component of the movement of the end-effector. The resulting closest point and normal are then transformed from model space into world space.

One obvious advantage to this approach is efficiency. If we assume there are n models with active surfaces and a total of m active surfaces then there will be $n + m \times 2$ points transformed. Like the second method, these transformations must be computed for each cycle through DPT. However, since the number of surfaces in a model usually is far less than the number of control points in all of the surfaces, this approach is more efficient than either of the other two approaches.

5.2.3 Contact and Tracing.

Contact is initiated when the penetration depth of the closest active surface becomes larger than zero, with penetration being calculated by projecting the end-effector onto the surface normal. Once contact has been established, this surface is considered *current* and surface tracing begins. The surface remains current until the end-effector either transitions off the model or onto an adjacent surface.

5.2.4 Transitions.

We use topological adjacency information contained in a solid model for surface transitioning. This information is most often available in CAD models and both simplifies and accelerates the transitioning problem. This is also a necessary first step to allowing transitions across models composed of trimmed NURBS surfaces.

Our method makes use of adjacency information provided by the *Alpha_1* modeling package. All of the closest points are tracked using the end-effector position. When the current surface's tracked point hits an edge, the adjacency table is queried to find the neighboring surface. If the closest point on the adjacent surface is not on an edge, then this surface is made current and tracing continues. Conversly, if the pair of closest points are both on the edge, special care must still be taken to determine if the user remains in contact with the model.

5.3 Physically Based Haptics

Modeling the physically based motion of models and assemblies adds to the realism of the virtual design environment. Models within the *Alpha_1* modeling environment can have attributes attached that indicate model properties such as inertia tensor, mass, and center of mass. In addition to these attributes, assembly information can also be provided to facilitate dynamic simulation.

5.3.1 Surface Interactions.

We are researching both forms of the response model. For the compliance model, a mapping of the Sarcos Arm torque sensors to force is used to calculate the force inputs. Once contact has been established with a virtual model by DPT, a control strategy renders the resulting acceleration to the user as well as the virtual object.

Our current system uses the stiffness model. A nonlinear surface response model (Marhefka and Orin, 1996) has been adopted to provide a physically-accurate model of probe-surface collision (Eq. 4). This model is given by the equation

$$F_{resp} = -kx^n - \lambda x^n \dot{x}, \tag{4}$$

where x is the penetration depth, $\dot{x}$ is the velocity of the end-effector, k is the spring coefficient, and λ is the damping coefficient. Notice that penetration depth is present in the second term of Eq. (4), which requires that force starts from zero during initial impact regardless of the velocity. We have found experimentally that $n = \frac{1}{2}$ is a sharp and stable value for the Sarcos Arm.

We are attempting to increase the stability of our arm during tracing to achieve smaller penetration depth. Numerically deriving velocity, $\dot{x}$, must be done carefully to avoid noisy chattering along the surface. A formulation that approximates velocity as good, on average, as the Kalman filter is

$$\dot{x} = \frac{1}{nT}(x_i - x_{i-n}), \tag{5}$$

where T is the sampling period and n is the historical sampling offset (Spong and Jaritz, 1996).

For our experiments, the value of n depended on sampling rate and noise attenuation. Experimentally, a value of $n = 15$ at 1500Hz sampling rate was found to be sufficient and nearly delay-free. As a basis for comparison, the Chebychev and Butterworth filters, when used in conjunction with finite difference operations, result in a delay of three times that of Eq. (5). The calculation of acceleration requires a second pass over the filters which makes the delay even more apparent.

Equation (5) not only compares favorably to the low pass filters above, but also to ideal differentiators. An appropriate discrete-time approximation for the frequency response of an ideal differentiator is $H_d(e^{j\omega}) = j\omega e^{-j\omega/2}$. The corresponding impulse response is

$$h_d(n) = \frac{(-1)^n}{\pi(n-1/2)^2}, \tag{6}$$

as described in (Jackson, 1996). The plot of this frequency response closely follows the continuous-time ideal differentiator. However, discrete ideal differentiators have a delay similar to the Chebychev or Butterworth approaches and are typically noise-sensitive.

In order to properly transmit forces to the appropriate actuators, we have formulated the wrench for the arm upon impact. The following equations allow the use of a tool frame for the finger without changing the Jacobian.

$$W = \begin{bmatrix} f_1 \\ r_1 \times f_1 \end{bmatrix}, \tag{7}$$

$$J = \begin{bmatrix} z_0 \times b_0 & \ldots & z_4 \times b_4 & 0 & 0 & 0 \\ z_0 & \ldots & z_4 & z_5 & z_6 & z_7 \end{bmatrix}, \tag{8}$$

$$\tau = J^T W, \tag{9}$$

where W is the wrench at the wrist, f_1 is the force at the finger tip, r_1 is the moment arm of the finger on the wrist, z_i is the axis of rotation for joint i, b_i is a vector from the origin of joint i to the wrist, τ is the torque vector, and J is the Jacobian.

5.3.2 Movable Model Interactions. We have developed test cases to rigorously test our moving surface and tracing dynamics. These tests cases are as follows:

- *Rigid Model.* The operator moves and/or rotates a model by pressing against its surface. The force returned from the wall model calculation and the models mass are used to calculate an acceleration for the model. The object motion is derived from numerical integration of this acceleration.
- *Push Button with Detent.* Upon contacting the push button, the operator feels the surface response in Fig. 6.

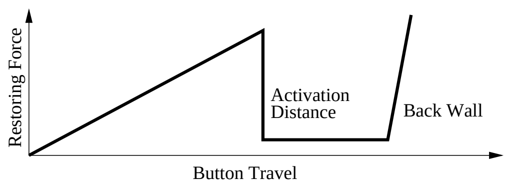


Figure 6. Force response model of a push button with detent.

Because the force response of the push button is due to surface penetration and the detent response model, we use a double spring to describe this interaction. The mass of the push button is assumed to be small. Letting the constants k_1 and k_2 be the spring coefficients of the surface penetration model and detent model respectively Eq. (10) solves for the force response and Eq. (11) solves for the position of the push button.

$$F = \frac{k_1 k_2}{k_1 + k_2} x_{travel}, \tag{10}$$

$$x_1 k_1 = x_2 k_2 = F, \tag{11}$$

where x_{travel} is a measured offset from the push button original position, x_1 is the penetration into the push button, and x_2 is the movement of the push button.

- *Pendulum.* An oscillating pendulum provides an excellent test case for a model that has constrained physically based movement. The angular acceleration $\ddot{\theta} = -mass * gravity * length * sin(\theta)/Inertia$ due to gravity is integrated to give angular velocity and position. The non-linear response model is used to generate user forces that are applied to the pendulum. Impulse forces during impact with the moving pendulum are assumed to be small and are ignored.

6 RESULTS

One measure of the quality of a haptic rendering is the amount of penetration depth into the model. However, having a small average penetration depth is not sufficient to demonstrate a smooth tracing experience. Penetration depth must also be shown to be consistently near the mean.

To demonstrate the smoothness and consistency of our tracing algorithm we performed two tests on five different models (Table 1). The models were given a mass of $40Kg$ and filled much of the usable workspace, about $1m^3$ for the Sarcos arm. The exception to this is the push button which was assumed to have a very low mass.

Model	Test	Mean	STD	Max
Cube	1	1.12	0.71	5.58
	2	2.05	1.59	8.18
Cylinder	1	1.93	1.16	6.11
	2	1.14	1.42	11.23
Goblet	1	2.40	1.79	9.30
	2	0.68	0.51	3.41
Pendulum	1	1.93	1.39	9.84
	2	2.79	1.79	9.10
Button	1	1.30	0.85	4.75
	2	1.00	0.53	3.18

Table 1. Analysis of penetration depths for surface tracing tests on five models. All measurements are in mm.

The first test case serves as a base line and consists of tracing the model in a static position. The second test case used moving models controlled by the physically based simulation; only values during contact were collected for analysis with each test consisting of over 10,000 data points. The Sarcos Arm was run with surface stiffness of 6000N$/m$ to provide a comfortable tracing experience.

Table 1 shows that the results for test 1 and test 2 are similar. The first column shows that the mean penetration depth in all cases was less than $3mm$ with an average mean of $1.63mm$. The standard deviation illustrates that the penetration depth is not only consistently near the mean, but also consistently small. The final column gives the maximum penetration depth observed during the test; this value usually occurs when the user either presses very hard against a static object or tries to move an object swiftly. Even in these cases the maximum value was below $12mm$ and on the average it was less than $7mm$.

To further illustrate the consistency of the trace algorithm, histograms are given in Fig. 7 for tests one and two as performed on the cylinder. Both histograms show a tight distribution of penetration depths with peaks at less than $2mm$.

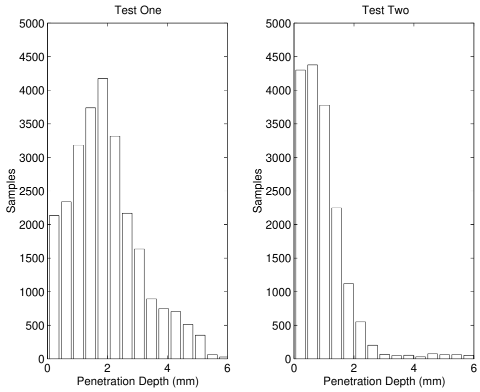


Figure 7. Penetration depth histograms for the cylinder.

Figure 8 illustrates the accuracy of our two-spring model for the push button with detent. The data was collected while the user depressed the button. The realized force was derived by mapping the Sarcos Arm load cells through the manipulator Jacobian pseudo-inverse.

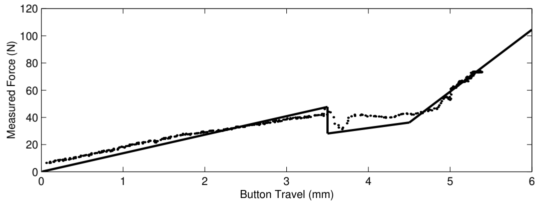


Figure 8. Measured force versus button travel.

7 CONCLUSION

We have demonstrated new techniques that allow virtual haptic environments to be populated by maneuverable NURBS models. These techniques, in conjunction with equations we have presented that produce high quality results with our Sarcos Dextrous Master (a high inertia device with a complex dynamics structure), increase the level of physically based interactivity. This combination greatly improves the amount of information a designer can gather about a design. Furthermore, with the synchronized visual and haptic display, the designer can immerse himself in the design environment.

8 ACKNOWLEDGMENTS

The authors would like to thank Dave Johnson for his help in editing the paper, Rod Frier for his help in setting up the Sarcos Arm and its various control systems, as well as the networking software and Ali Nahvi for his help with the wrench formulations. Thanks also go to the students and staff of the *Alpha_1* project, within which this work was developed. Support for this research was provided by NSF Grant MIP-9420352, by DARPA grant F33615-96-C-5621, and by the NSF and DARPA Science and Technology Center for Computer Graphics and Scientific Visualization (ASC-89-20219).

REFERENCES

Adachi, Y., "Touch And Trace On The Free-Form Surface Of Virtual Object," in *Proc. Virtual Reality Annual Intl. Symp.,* Seattle, WA, pp. 162-168, September, 1993.

Adachi, Y., Kumano, T., and Ogino, K., "Intermediate Representation For Stiff Virtual Objects," in *Proc. Virtual Reality Annual Intl. Symp.,* Research Triangle Park, NC, pp. 203-210, March 11-15, 1995.

Cohen, E., Lyche, T., and Riesenfeld, R., "Discrete B-Splines And Subdivision Techniques In Computer Aided Geometric Design And Computer Graphics," *Computer Graphics and Image Processing,* Vol 14, Number 2, October 1980.

Colgate, J.E., and Brown, J.M., "Factors Affecting The Z-Width Of A Haptic Display," in *Proc. IEEE 1994 International Conference on Robotics & Automation,* pp. 3205-10, San Diego, CA, 1995.

Hollerbach, J.M., Cohen, E., Thompson, W.B., and Jacobsen, S.C., "Rapid Virtual Prototyping Of Mechanical Assemblies," *NSF Design and Manufacturing Grantees Conference,* Albuquerque, NM, Jan. 3-5, 1996.

Hollerbach, J.M., Cohen, E., Thompson, W.B., Freier, R., Johnson, D., Nahvi, A., Nelson, D., Thompson II, T.V., and Jacobsen, S.C., "Haptic Interfacing For Virtual Prototyping Of Mechanical CAD Designs," *ASME Design for Manufacturing Symposium,* Sacramento, CA, Sept. 14-17, 1997,

Jackson, L., *Digital Filters and Signal Processing,* Kluwer Academic Publishers, pp. 151-152, 1996.

Jacobsen, S.C., Smith, F.M., Iversen, E.K., and Backman, D.K., "High Performance, High Dexterity, Force Reflective Teleoperator," in *Proc. 38th Conf. Remote Systems Technology,* Washington, D.C., pp. 180-185, November, 1990.

Johnson, D.E., and Cohen, E., "Minimum Distance Queries For Polygonal and Parametric Models," Technical Report UUCS-97-003, University of Utah, Department of Computer Science, Feb. 26, 1997.

Marhefka, D.W., and Orin, D.E., "Simulation Of Contact Using A Nonlinear Damping Model," in *Proc. International Conference on Robotics and Animation,* Minneapolis, Minnesota, pp. 1662-1668, April 1996.

Mark, W.R., Randolph, S.C., Finch, M., Van Verth, J.M., and Taylor III, R.M., "Adding Force Feedback To Graphics Systems: Issues And Solutions," in *Proc. SIGGRAPH 96,* New Orleans, pp. 447-452, August. 4-9, 1996.

Minsky, M., Ouh-Young, M., Steele, M., Brooks, F.P. Jr., Behensky, M., "Feeling And Seeing: Issues In Force Display," in *Proc. Symposium on Interactive 3D Graphics,* Snowbird, Utah, pp. 235-243, 1990.

Riesenfeld, R., "Design Tools For Shaping Spline Models," in *Mathematical Methods in Computer Aided Geometric Design,* (Edited by T. Lyche and L. Schumaker), Academic Press, 1989

Riesenfeld, R., "Modeling With Nurbs Curves And Surfaces," in *Fundamental Developments of Computer Aided Geometric Design,* L. Piegl (ed.), Academic Press, 1993.

Salisbury, J., Brock, D., Massie, T., Swarup, N., and Zilles, C., "Haptic Rendering: Programming Touch Interaction With Virtual Objects," in *Proc. Symposium on Interactive 3D Graphics,* Monterey, CA, pp. 123-130, 1995.

Snyder, John, "An Interactive Tool For Placing Curved Surfaces Without Interpenetration," in *Proc. SIGGRAPH 95,* Los Angeles, pp. 209-218, August. 6-11, 1995.

Spong, M., Jaritz, A., "An Experimental Comparison Of Robust Algorithms On A Direct Drive Manipulator," in *IEEE Transactions on Control Systems Technology,* Vol. 4, No. 5, November 1996.

Thompson II, T.V., Johnson, D.E., Cohen, E.C., "Direct Haptic Rendering Of Sculptured Models," in *Proc. Symposium on Interactive 3D Graphics,* Providence, RI, April 27-30, 1997.

Yoshikawa, T., Yokokohji, Y., Matsumoto, T., Zheng, X., "Display of Feel for the Manipulation of Dynamic Virtual Objects," in *Journal of Dynamic Systems, Measurement, and Control,* Vol. 117, December 1995.

Zilles, C.B., and Salisbury, J.K., "A Constraint-Based God-Object Method For Haptic Display," in *Proc. IEE/RSJ International Conference on Intelligent Robots and Systems, Human Robot Interaction, and Cooperative Robots,* Vol 3, pp. 146-151, 1995.