Haptic Rendering of Planar Rigid-Body Motion using a Redundant Parallel Mechanism

D. Constantinescu, I. Chau, S. P. DiMaio, L. Filipozzi, S. E. Salcudean, and F. Ghassemi
Department of Electrical and Computer Engineering,
University of British Columbia, Vancouver, BC, V6T 1Z4, Canada
{*tims@ee.ubc.ca*}

Abstract

We present a system for rendering planar rigid-body motion by means of a redundant parallel mechanism. The device design, the control architecture and the passive virtual environment simulation are presented. The system is used to compare various virtual walls and friction models proposed for haptic applications. In addition, the reset-integrator dry friction model proposed by Haessig and Friedland [8] is implemented in a haptic interface for the first time.

1 Introduction

Haptic displays provide tactile and kinesthetic feedback to a user interacting with a virtual environment. Potential applications of these devices (CAD, medical, training, etc) require physically meaningful user interaction with a complex dynamic world. Such interaction poses challenges for both device and virtual world design. The device should have a large, isotropic workspace, high spatial acceleration and resolution, low inertia and friction. The simulation should be a real-time, passive, general-purpose, physics-based simulation. In conjunction, these characteristics ensure a high quality interface, decoupling of the device controller design from the simulation design, and flexibility of the virtual environment.

There are few reports of rigid body interaction with virtual environments [4, 1]. Most published research has addressed haptic primitives (virtual walls [15, 7], textures [16], friction [15, 14, 10], inertial dynamics [18]) or point interaction with static environments [19]. More recently, haptic rendering of rigid-body motion has been reported in [4] and [1].

Chang and Colgate [4] have discussed the hard real-time constraints that haptic applications impose on the virtual world. They met these constraints by interfacing the device with an impulse-based simulation. The appeal of this approach is that it does not require the modeling of constraints and it can be made passive. Its adaptation to haptic interaction, however, encounters difficulties in determining a valid impact state and in rendering friction. These difficulties arise because the impulse-based method does not allow body interpenetration, while the haptic simulation requires a fixed-step size integration routine and, therefore, cannot avoid it.

To meet the haptic controller requirements, Berkelman *et al* [1] have chosen an architecture where the controller and the simulation run independently on separate processors. This alleviates the hard real time constraints on the simulation and forces the controller to interpolate simulation data. The price paid is a feeling of stickiness or sluggishness in the interaction [1]. Nevertheless, this implementation demonstrates the most complex rigid body interaction with a virtual world at the present time.

In this paper, we present a haptic interface where the controller and the simulation run synchronously. The device has a large motion range and renders planar motion. The haptic update rates are accommodated by a general-purpose penalty-based simulation which uses passive numerical methods. The paper is organized as follows: the device design and the control architecture are overviewed first; the collision detection and the dynamic system implementation are discussed next; then, various virtual walls and dry friction models are surveyed and compared. Finally, conclusions and recommendations for future work are presented.

2 Haptic Interface Design and Control Architecture

A virtual environment system serves as a test bed for the concepts presented in Section 4. It comprises a haptic interface, virtual slave and environment models, a controller that coordinates both force and position information between the haptic interface and the virtual environment, and a graphical display, as depicted in Figure 1.

The haptic interface has three degrees of freedom allowing for planar translation and unlimited rotation about a single axis. This is achieved by using a dual panto-

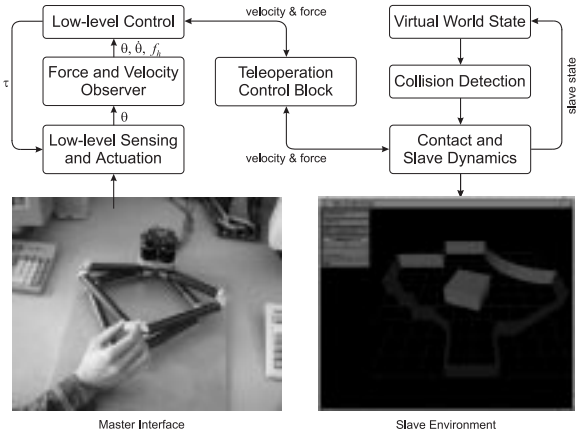

Figure 1: Virtual environment system

graph arrangement - also shown in Figure 1. Each pantograph is driven by two DC motors located at the base joints, while their endpoints are coupled by means of a linkage, to which the interface handle is connected. This linkage forms a crank that allows unlimited rotation of the handle. The position of the handle, as well as applied hand forces and torques are estimated using only joint angle measurements, applied motor torques and a detailed model of the mechanism dynamics [17]. A low-level impedance controller governs the device dynamics and is coupled to the slave environment via a teleoperation controller [17]. The virtual slave is a rectangular block contained within an enclosure of virtual walls. A collision detection algorithm identifies and computes contacts between virtual slave and environment.

From the programmer's perspective, the operating environment consists of two computers, an SGI running IRIX and an Intel Pentium-based personal computer running QNX4. The two computers communicate via a local area network and the haptic device is interfaced to the personal computer via a multifunction I/O adapter for joint angle sensing and actuation.

The SGI displays the virtual environment while the QNX4 computer runs the haptic control architecture. Composed of four operating processes, the control architecture is responsible for communicating commands and feedback values with the virtual environment display, for executing the four-channel haptic controller and for computing the dynamic response of the slave device at a control loop sampling rate of 500Hz.

3 Collision Detection

The collision detection is the integration of several well-known algorithms and methodologies. It adopts a two-level architecture to limit computation time. The bounding box method and a fast distance computation algorithm are used to prune the objects that cannot possibly collide. The GOD-object model [19] is then adapted to handle volumetric contact. These components are discussed in detail below.

The Two-Level Architecture

The collision detection consists of two processes: the global proximity detection and the local contact detection. The global proximity detection identifies the objects that are close enough for collision and reports these potential collision pairs to the local contact detection process. The frequency of the local detection is that of the control loop. The frequency of the global detection is much lower, since it is computationally expensive to check every pair of virtual objects in every control loop. Moreover, the virtual world changes, in the global sense, less frequently.

Global Proximity Detection

This module uses the distance between a pair of objects to decide whether they are likely to collide. When this distance is smaller than a threshold, the pair is declared a potential collision pair and is reported to the local contact detection module.

The proximity detection uses the constant complexity distance computation algorithm developed by Cameron [3], which also returns additional useful information, such as the closest features between two objects. Computational cost is further reduced by using the axis-aligned bounding box method to filter out pairs which cannot possibly collide [11].

Local Contact Detection

This process computes the closest features, the distance between the closest features, and the contact point of a potential collision pair reported by the global proximity detection. The closest features are a vertex and an edge from each object, respectively. Every pair is checked in each control loop. When the distance exceeds a threshold, the pair is discarded to reduce unnecessary workload.

Since the objects are free to move, their closest features can change. Nevertheless, all polygons are convex, so their new closest features must be neighbors of the current ones. Moreover, at most two closest features exist for each pair of objects at any moment. The search for a new contact can be done in constant time.

In addition, we need the trajectory "history" to find the accurate contact point, as difficulties are reported when using only the closest features [19]. We resolve these problems by implementing a GOD-object [19] for each contact.

The vertex-edge (VE) contact is the basic building block for the different types of contacts. A vertex-vertex contact is modeled as two VE contacts. This is illustrated in Figure 2. As the upper body moves along the edge, a new VE contact occurs when the vertices are close

to each other. This additional vertex penetration will generate a force in the simulator discussed in Section 4, but it is not noticeable because of its small magnitude. Similarly, edge-edge contacts are modeled by pairs of VE contacts.

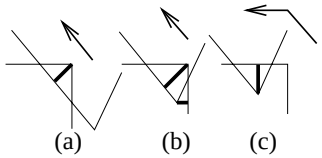

Figure 2: Vertex-vertex contact modeled as two vertex-edge contacts when the upper body moves along its edge and contact topology changes.

4 Dynamic System Implementation

The simulation receives the forces generated by the hand and the controller, computes the environment reaction forces and advances the state of the virtual object using a fixed step-size integration routine. The inertial properties are modeled for the rigid body as a whole while normal and frictional reaction forces are modeled only at the vertices with edge-edge contact represented by two VE contacts.

To compute the environment reaction, we use the method proposed by Salcudean and Vlaar [15] which includes a frictionless collision impulse upon contact and an interaction force during contact. Hence, each collision pair reported by the collision detection algorithm is viewed as being in one of three possible contact states: no contact, colliding contact, or continuous (sticking or sliding) contact, as depicted in Figure 3. State transitions occur based on local contact information.

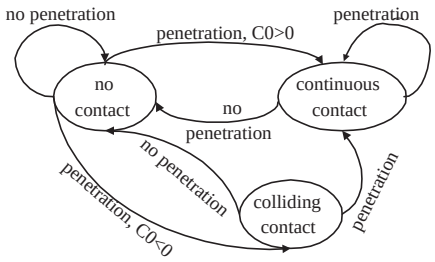

Figure 3: The contact states of one collision pair.

Passivity is the primary goal in designing the dynamic simulator. This makes a strictly passive controller a sufficient condition for stable interaction. Passivity is achieved by resolving each contact state using discrete-time passive numerical methods (i.e. obeying the principle of conservation of energy [2]).

Due to the constant step size of the integration routine, invalid impact states (bodies departing from each other when a new contact is declared) and multiple collisions can occur in a rigid body simulation. Rather than searching for a more accurate collision time, which is computationally expensive, or artificially modifying the impact state [4], we choose to bypass invalid impact states by declaring continuous contact, as shown in Figure 3. Continuous contact is a physically motivated approximation of colliding contact.

Multiple impacts are treated as simultaneous collisions [9]. Relative velocity increments at each contact are computed based on Poisson's restitution hypothesis and post-impact body velocities are determined by imposing contact velocity constraints.

During continuous contact, interaction forces have a normal component, modeling contact stiffness, and a tangential component, modeling dry friction. The normal component is calculated as:

$$\begin{aligned} F_{n,k} &= -K_c\tilde{d}_k - B_c C_{0,k} \quad , \\ \tilde{d}_k &= 2d_k - 1.5d_{k-1} + 0.5d_{k-2} \end{aligned} \tag{1}$$

where F_n is the normal contact reaction, K_c and B_c are the contact stiffness and damping, C_0 is the relative compression velocity, d is the penetration depth, and k is the time step. Contact passivity is ensured even in the absence of damping by the "sample-estimate-hold" (SEH) [6] value of the penetration depth, $\tilde{d}$.

Dry friction is rendered using the reset-integrator model proposed by Haessig and Friedland [8]:

$$F_f = \begin{cases} (1+a)K_r p + \beta S_0 & \text{if stick} \\ K_r p & \text{if slip} \end{cases} \tag{2}$$

where:

$$\text{slip} \iff \begin{cases} S_0 > 0 & \text{and} \quad p \geq p_0 \\ & \text{or} \\ S_0 < 0 & \text{and} \quad p \leq -p_0 \end{cases}$$

$$\text{stick} \qquad \text{otherwise.}$$

In (2), p is the strain of the bond, K_r is the spring rate, a is the "stiction gradient", S_0 is the sliding velocity, and β is a damping coefficient. Thus, the contacting points on the two bodies are connected by a spring with a spring constant of $(1+a)K_r$ during sticking and K_r during sliding. When the strain exceeds a threshold p_0 and it tends to increase further, the bodies slide over each other. Otherwise, they stick. The oscillations occurring when entering into the sticking mode are damped by the βS_0 term.

Free motion passivity is ensured by using the passive numerical integrator discussed in [2] along each DOF.

5 Survey of Haptic Contact Models

The haptic interface presented above renders rigid body inertia and frictional contact. In the following, the con-

tact model that we use is compared through simulations and experiments with other proposed models.

Virtual Walls

The simulation technique that we have adopted computes contact forces proportional to a constant contact stiffness. For haptics, this has two drawbacks: (i) it cannot render infinite stiffnesses; and (ii) its zero order hold ZOH implementation is active [5]. Proposed virtual walls address both of these problems.

Minsky *et al* [13] used continuous time control theory to analyze the ZOH wall. They suggested passivating the wall by increasing its damping for given stiffness and sampling rate. However, Colgate *et al* [5] showed that it was the discretization that destroyed the passivity of the pure stiffness. They proved that a passive ZOH wall required physical damping in the device and that increasing the virtual damping was detrimental to stability.

Ellis *et al* [6] devised a predictor-corrector force discretization that satisfied the principle of energy conservation more accurately.

Goldfarb and Wang [7] proposed that the energy introduced by the ZOH be dissipated by means of dry friction. Their wall consists of springs in parallel, each attached to an inertial block with Coulomb friction. As the contact force increases, the blocks start sliding and energy is lost to friction. Energy loss (hysteresis curve of the wall) is adjusted through block masses and coefficients of friction. Thus, wall passivity is guaranteed independent of other simulation parameters.

For increased perceived stiffness, Salcudean and Vlaar [15] used a braking pulse upon wall penetration which resulted in higher damping on the wall surface, while Massie [12] introduced a term analogous to the integral action of a PID controller which increased wall stiffness during prolonged contact.

Simulation results for a rectangular object pushed by a constant force into undamped walls are plotted in Figure 4. The results show that the SEH marginally passivates a pure stiffness and it may or may not passivate a PI wall, depending on the choice of the integral component. Most effective in achieving contact stability, is the wall with pulse. The hysteretic wall can closely follow its performance by an appropriate choice of parameters.

Experimental results obtained after implementing the walls on our haptic interface are plotted in Figures 5-6. In the experiment, the user holds a rectangular virtual object inside a frictionless rectangular world. The user lets go of the handle and the hand is replaced by a constant force. No wall damping exists. Unlike in the simulation, initial contact with the wall is not an edge-edge contact, due to the user and controller action, as

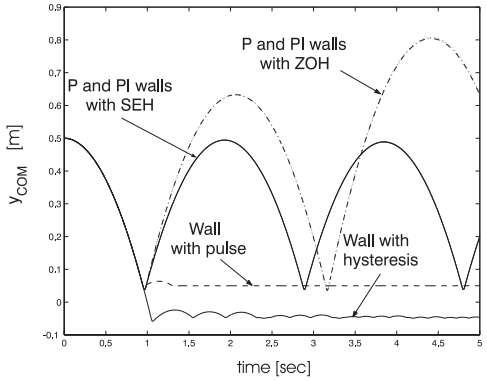

Figure 4: Rectangular object bouncing against various walls (simulation).

well as the interface drift. Nevertheless, the experiment shows that contact stability improves as the passivity of the wall increases. As depicted in Figure 5, the object bounces continuously against the marginally passive or active walls. Contact stability is achieved upon the first impact with the wall with pulse and is quickly achieved for the wall with hysteresis (Figure 6). The advantage of the wall with pulse over the wall with hysteresis is that its restitution properties are more easily and intuitively adjusted. This requires choosing one coefficient of restitution instead of choosing several masses and coefficients of friction.

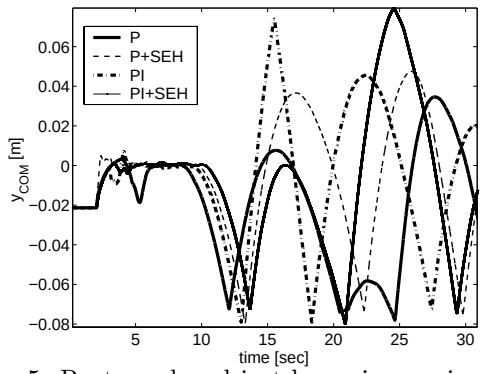

Figure 5: Rectangular object bouncing against marginally passive and active walls (experiment).

Friction Models

Haptic implementations of Coulomb friction have been reported for point masses: a modified Karnopp model (with viscous friction in sliding) by Salcudean and Vlaar [15], a modified Dahl model (drift-free) by Hayward and Armstrong [10], and a model based on human finger pad characteristics by Nahvi *et al* [14].

For rendering frictional body contact, we use the reset-integrator model proposed by Haessig and Friedland [8]. The modified Dahl model, the model based on human finger pad characteristics and the classical model [8]

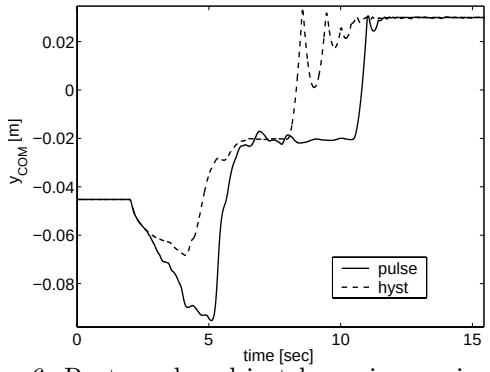

Figure 6: Rectangular object bouncing against passive walls (experiment).

have also been implemented on the body vertices, since they require only state information (position and velocity) regarding the local contact. In the following, these models are compared by means of experiments.

All models recognize two states of a frictional contact: slip and stick. State transitions occur based on the relative sliding velocity in the classical model. The other models represent stiction as a compliance, with the peak stiction force corresponding to the peak bond strain. Transition from stick to slip is driven by the strain value, while transition from slip to stick is driven by the contact sliding velocity. Simulation results for a peg pushed in a tight hole by a sinusoidal unidirectional force F are plotted in Figure 7. The wall normal reaction is implemented as a pure stiffness with predictor-corrector force discretization.

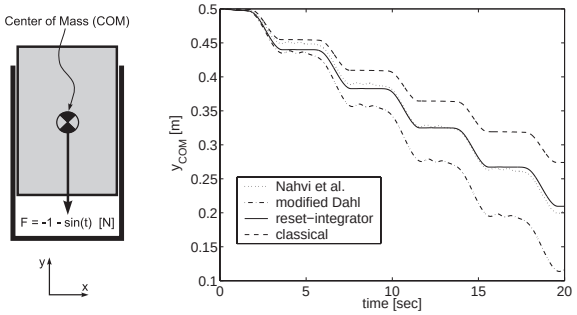

Figure 7: Position of the center of mass of a peg pushed in a hole with friction (simulation).

Figures 8-11 show the same scenario implemented on the interface. Initially, the user pushes the peg into the hole. When the user releases the handle, the hand force is replaced by the sinusoidal unidirectional force. Both simulation and experiment confirm that all models exhibit slip and stick. However, the classical model allows the body to accelerate even when the applied force is less than the peak stiction force: it exhibits drift that the user can feel. The driftless Dahl model and the model based on human finger pad characteristics introduce perceivable compliance during the stick phase. This might not be desirable for rendering frictional contact between rigid bodies. Thus, the reset-integrator model is considered most suited for kinesthetic feedback applications. Its damping factor allows an extra degree of freedom in designing the contact characteristics: the higher the damping, the crisper the transition between slip and stick.

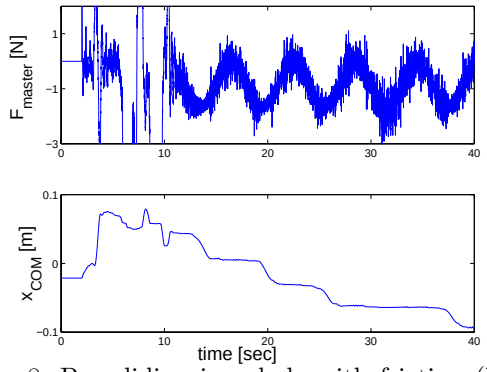

Figure 8: Peg sliding in a hole with friction (Nahvi *et al* [14]).

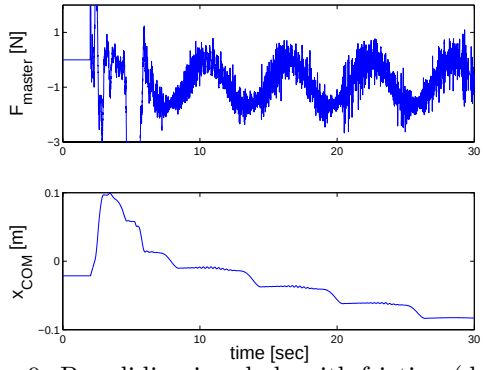

Figure 9: Peg sliding in a hole with friction (driftless Dahl [10]).

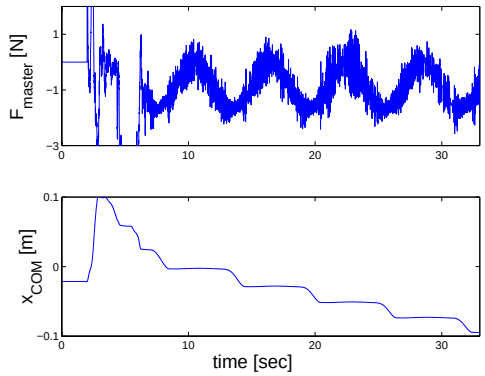

Figure 10: Peg sliding in a hole with friction (reset-integrator).

6 Conclusions

We have developed a system that allows the user to interact with a planar world, while observing both visual and kinesthetic feedback. The haptic controller

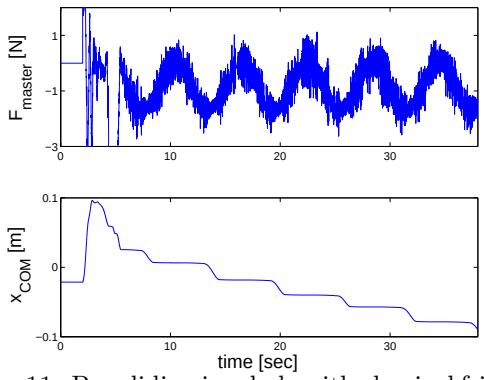

Figure 11: Peg sliding in a hole with classical friction.

and simulation designs have been decoupled by a passive implementation of the virtual world. The simulation computes environment reaction either as penalty contact forces or frictionless collision impulses. Incorporation of friction in the treatment of simultaneous collisions is under investigation.

The system has been used to compare different virtual walls and friction models in terms of their passivity and rendering of the slip-stick phenomenon, respectively. However, further studies should be conducted to compare these models perceptually.

In the present implementation, the simulation uses only observer-based force information from the haptic device. Future work will investigate the use of both position and force information.

Acknowledgments

This work has been supported by the Faculty of Graduate Studies at UBC and by the Canadian IRIS/PRECARN Network of Centers of Excellence.

References

[1] P.J. Berkelman, R.L. Hollis, and D. Baraff. Interaction with a Realtime Dynamic Environment Simulation using a Magnetic Levitation Haptic Interface Device. In *Proc. IEEE Int. Conf. on Robot. and Aut.*, pages 3261–3266, Detroit, MI, 1999.

[2] J.M Brown. Passive Implementation of Multibody Simulation for Haptic Display. *Ph.D. thesis, Northwestern University*, 1998.

[3] S. Cameron. Enhancing GJK: Computing Minimum and Penetration Distances between Convex Polyhedra. In *Proc. 1997 IEEE Int. Conf. on Robot. and Aut.*, pages 3112–3178, Albuquerque, NM, 1997.

[4] B. Chang and J.E. Colgate. Real-Time Impulse-Based Simulation of Rigid Body Systems for Haptic Display. In *Proc. ASME, Dynamic Systems and Control Division*, pages 1–8, Houston, TX, 1997.

[5] J.E. Colgate, P.E. Grafing, M.C. Stanley, and G. Schenkel. Implementation of Stiff Virtual Walls in Force-Reflecting Interfaces. In *Proc. IEEE Virt. Real. Ann. Int. Symp.*, pages 202–208, Seattle, WA, 1993.

[6] R.E. Ellis, N. Sarkar, and M.A. Jenkins. Numerical Methods for the Haptic Presentation of Contact: Theory, Simulations, and Experiments. In *Proc. AMSE, Dynamic Systems and Control Division*, volume DSC-58, pages 413–420, New York, NY, 1996.

[7] M. Goldfarb and J. Wang. Passive Stiffness Simulation with Rate-Independent Hysteresis. In *Proc. ASME, Dynamic Systems and Control Division*, volume DSC-67, pages 345–350, Nashville, TN, 1999.

[8] D.A.Jr. Haessig and B. Friedland. On the Modeling and Simulation of Friction. *J. Dyn. Syst. Meas. Contr.*, DSC-113:354–362, 1991.

[9] I. Han and B.J. Gilmore. Impact Analysis for Multiple Body Systems with Friction and Sliding Contact. In D.P. Sathyadev, editor, *Flex. Assembly Systems*, pages 99–108. New York, 1989.

[10] V. Hayward and B. Armstrong. A New Computational Model of Friction Applied to Haptic Rendering. In P. Corke and J. Trevelyan, editors, *Experimental Robotics VI, Lecture Notes in Control and Information Sciences*, pages 403–412. Springer-Verlag, 2000.

[11] M. Lin and J. Canny. A Fast Algorithm for Incremental Distance Calculation. In *Proc. Int. Conf. on Robot. and Aut.*, pages 1008–1014, Sacramento, CA, 1991.

[12] T.M. Massie. Taking the Mush Out of Haptics with Infinitely Stiff Walls. In *Proc. 1st PHANToM User's Group Workshop*, pages 17–19, Dedham, MA, 1996.

[13] M. Minsky, M. Ouh-young, O. Steele, F.P.Jr. Brooks, and M. Behensky. Feeling and Seeing: Issues in Force Display. *Comp. Graph.*, 24(2):235–243, 1990.

[14] A. Nahvi, J.M. Hollerbach, R. Freier, and D.D. Nelson. Display of Friction in Virtual Environments Based on Human Finger Pad Characteristics. In *Proc. ASME, Dynamic Systems and Control Division*, volume DSC-64, pages 179–184, Anaheim, CA, 1998.

[15] S.E. Salcudean and T.D. Vlaar. On the Emulation of Stiff Walls and Static Friction with a Magnetically Levitated Input/Output Device. *J. Dyn. Syst. Meas. Contr.*, 119:127–132, 1997.

[16] J. Siira and D.K. Pai. Haptic Textures - A Stochastic Approach. In *Proc. IEEE Int. Conf. on Robot. and Aut.*, pages 557–562, Minneapolis, MN, 1996.

[17] Mohammad R. Sirouspour, S. P. DiMaio, S. E. Salcudean, P. Abolmaesumi, and C. Jones. Haptic Interface Control – Design Issues and Experiments with a Planar Device. In *IEEE Int. Conf. on Robot. and Aut.*, San Francisco, CA, 2000.

[18] T. Yoshikawa, Y. Yokokohji, T. Matsumoto, and X.-Z. Zheng. Display of Feel for the Manipulation of Dynamic Virtual Objects. *J. Dyn. Syst. Meas. Contr.*, 117:554–550, 1995.

[19] C.B. Zilles and J.K. Salisbury. A Constraint-based God Object Method for Haptic Display. In *IEEE Int. Conf. Intel. Rob. and Syst.*, volume 3, pages 146–151, Piscataway, NJ, 1995.