A Modular Haptic Rendering Algorithm for Stable and Transparent 6-DOF Manipulation

Miguel A. Otaduy and Ming C. Lin, *Member, IEEE*

***Abstract*—This paper presents a modular algorithm for six-degree-of-freedom (6-DOF) haptic rendering. The algorithm is aimed to provide transparent manipulation of rigid models with a high polygon count. On the one hand, enabling a stable display is simplified by exploiting the concept of virtual coupling and employing passive implicit integration methods for the simulation of the virtual tool. On the other hand, transparency is enhanced by maximizing the update rate of the simulation of the virtual tool, and thereby the coupling impedance, and allowing for stable simulation with small mass values. The combination of a linearized contact model that frees the simulation from the computational bottleneck of collision detection, with penalty-based collision response well suited for fixed time-stepping, guarantees that the motion of the virtual tool is simulated at the same high rate as the synthesis of feedback force and torque. Moreover, sensation-preserving multiresolution collision detection ensures a fast update of the linearized contact model in complex contact scenarios, and a novel contact clustering technique alleviates possible instability problems induced by penalty-based collision response.**

***Index Terms*—Clustering, collision detection, haptic rendering, numerical integration.**

I. Introduction

HUMANS USE tactile and force cues to explore the environment around them and to identify and manipulate objects. The synthesis of force and torque feedback arising from object–object interaction, commonly called six-degree-of-freedom (6-DOF) haptic rendering, can greatly benefit many applications involving dexterous manipulation and complex maneuvering of virtual objects. Examples of such applications include assembly and disassembly operations in rapid prototyping [1], [2] and endoscopic surgical training [3], [4].

Six-DOF haptic rendering is, in essence, an interactive computational process, and it comprises three main tasks: the computation of the position and orientation of a virtual tool manipulated by the user, the execution of collision detection and contact response between the tool and other objects, and the synthesis of force and torque that are displayed back to the user. The quality of haptic rendering can be measured in terms of the dynamic range of mechanical impedances that can be simulated stably [5]. The ability to render low impedance in free-space motion and high impedance during contact with stiff objects can also be regarded as the transparency or responsiveness of the haptic rendering system [6].

The focus of this paper is the design of a 6-DOF haptic rendering algorithm for rigid polygonal models with (typically nonconvex) complex geometry. The key to stable and transparent rendering is a very high force update rate [5], [6], but achieving it becomes a challenging task in complex contact scenarios between objects with a high polygon count, due to the inherent cost of collision detection. Enforcing the stability of the display can be highly simplified through a *virtual coupling* that decouples the synthesis of interaction forces from the simulation of the virtual environment, provided that this simulation is guaranteed to be discrete-time passive [7]. However, our experiments show that, even if the display is stable, it may not reach the desired degree of transparency, suffering, for example, from excessive free-space forces. Much of the design effort of our rendering algorithm was aimed at maximizing the transparency of the haptic display and the responsiveness of the visual simulation, while trying to maintain discrete-time passivity of the simulation of the virtual environment and, thereby, the stability of the display.

The main contributions of our algorithm are as follows.

- A formulation of the haptic display founded on the rigid body dynamic simulation of the virtual tool, solved using semi-implicit backward Euler integration. We borrow the concept of virtual coupling [7] for decoupling the simulation of the virtual tool from the synthesis of force feedback, but our proposed backward differentiation of the coupling forces enables small tool mass values that facilitate transparent manipulation.
- A linearized contact model, inspired by the concept of *intermediate representation* [8], for enhacing the stability and responsiveness of the simulation. The linearized contact model decouples the simulation of the virtual tool from the execution of collision detection, and enables fast update and implicit integration of the contact forces, resulting in the use of high-contact stiffness values while maintaining stability of the simulation.
- A contact clustering algorithm based on K-means clustering that provides smoother contact information and limits the translational contact stiffness applied to the virtual tool.
- The integration with our fast, perceptually based multiresolution collision detection algorithm [9], which enables a frequent update of the linearized contact response with minimal perceptible errors.

Manuscript received December 21, 2005. This paper was recommended for publication by Associate Editor E. Papadopoulos and Editor K. Lynch upon evaluation of the reviewers' comments. This work was supported in part by the University of North Carolina through a Computer Science Alumni Fellowship, in part by the Swiss National Science Foundation, in part by the U.S. National Science Foundation, in part by the Office of Naval Research, in part by the Army Research Office, in part by the Defense Advanced Research Projects Agency, in part by RDECOM, and in part by Intel Corporation.

M. A. Otaduy is with the Computer Graphics Laboratory, ETH-Zurich, CH-8092 Zurich, Switzerland (e-mail: otaduy@inf.ethz.ch).

M. C. Lin is with the Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-3175 USA (e-mail: lin@cs.unc.edu).

Digital Object Identifier 10.1109/TRO.2006.876897

- Stable and transparent 6-DOF haptic rendering of polygonal models with tens of thousands of triangles. We present experiments of interactive haptic manipulations using our proposed algorithm to evaluate the impact of the different modules on the overall stability and transparency.

The remainder of this paper is organized as follows. Section II discusses related work, and Section III presents an overview of the rendering algorithm. Sections IV–VI describe the implicit integration of rigid body simulation, the formulation of virtual coupling, and collision detection and response. Section VII presents the results. To conclude, Section VIII summarizes our work and discusses future research directions.

II. Related Work

Here, we summarize important findings on the stability of haptic display that have driven the design of the rendering algorithm. We also discuss general existing approaches for 6-DOF haptic rendering, as well as particular techniques for collision detection and collision response.

A. Stability of Haptic Rendering

Early stability analysis in haptic rendering focused on the problem of rendering stiff virtual walls. Several researchers reached the conclusion that high force update rates are necessary in order to achieve stable rendering [5], [6], [10]. Intermediate representations [8] have been very successful at maximizing the update rate of haptic rendering systems by performing a full update of the virtual environment at a low frequency (limited by computational resources and the complexity of the system) and using a simplified approximation for performing high-frequency updates of force feedback.

A number of techniques for 6-DOF haptic rendering follow the approach of *direct rendering* [11]–[13]. In direct rendering, the virtual tool follows rigidly the position of the haptic device, and collision forces are displayed directly. In this way, there is no need to simulate the dynamics of the virtual tool, allowing potentially for a highly transparent display. However, guaranteeing stable and responsive display is a daunting task, as both the rotational impedance and the frame rate may be highly variable. Small contact stiffness values result in large, visually perceptible interpenetrations, while large contact stiffness values may induce instabilities when the frame rate of collision detection drops.

Colgate *et al.* [7] proposed a multidimensional viscoelastic virtual coupling for stable interaction with nonlinear virtual environments. If the implementation of the virtual environment is guaranteed to be discrete-time passive, the design of a stable display is reduced to appropriate tuning of the parameters of the coupling. Colgate *et al.* [7] also pointed out that one way of obtaining a discrete-time passive virtual environment is to implicitly integrate a continuous-time passive system. Adams and Hannaford [14] extended the concept of virtual coupling by providing a unifying framework for impedance and admittance displays. Several techniques for 6-DOF haptic rendering combine virtual coupling with rigid body simulation of the virtual tool [1], [15]–[17]. Wan and McNeely [2] instead employed a quasi-static simulation of the virtual tool. As mentioned in the introduction, the transparency of haptic display through virtual coupling may degrade with a slow simulation update rate or with a large virtual tool mass.

Researchers have also explored more flexible ways of ensuring system stability or passivity. Miller *et al.* [18] have extended Colgate's passivity analysis techniques, relaxing the requirement of passive virtual environments but enforcing *cyclo-passivity* of the complete system. Hannaford *et al.* [19] have investigated the use of passivity observers and passivity controllers. Mahvash and Hayward [20] have derived conditions for the passivity of a virtual environment where continuous-time passive local force models are activated sequentially. Following those conditions, they can design passive simulations of tool contact with deformable models.

B. Collision Detection for Haptic Rendering

The application of 6-DOF haptic rendering algorithms on complex models and complex contact scenarios presents several challenges. One of the fundamental challenges is the inherent cost of collision detection that results in slow force updates. McNeely *et al.* [1], and later Wan and McNeely [2], suggested solutions that discretize the objects at admissible resolutions, combining point-sampling and voxelization. Others have used acceleration data structures that exploit the rigidity of the geometry [11]–[13]. Recently, Johnson and Willemsen [13] have presented a fast, approximate, contact-point-tracking algorithm that is combined with slower exact collision updates. Otaduy and Lin [9] presented a sensation-preserving simplification technique that selects object resolutions adaptively at each contact. Otaduy *et al.* [21] have also proposed an algorithm to capture contact information between textured surfaces for 6-DOF haptic rendering.

Another challenge associated with complex models is the description of the contact manifold, which is commonly addressed by using multiple samples or contact points. A large number of contact points leads to expensive simulation with constraint-based approaches and causes instability problems with penalty-based collision response due to the increase of the total contact stiffness. McNeely *et al.* [1] suggested limiting the total stiffness after reaching a certain number of contacts, while Kim *et al.* [12] proposed a proximity-based clustering technique that reduces the number of representative contacts. Luo and Xiao [22] apply geometric and dynamic rules to determine a minimum set of active contacts, their configuration, and collision forces.

C. Collision Response Between Rigid Bodies

Three commonly used techniques exist for computing collision response between rigid bodies: constraint-based, impulse-based, and penalty-based. Early constraint-based techniques stopped the simulation at all collision events and formulated a linear complementarity problem to solve for collision forces and object accelerations. Some researchers have integrated this approach with haptic rendering [16], [17], but they have tested it only on relatively simple benchmarks, due to the typically high cost of variable time-stepping. Others have developed constraint-based techniques with fixed time-stepping [23], but they may suffer from drift since the constraints are expressed on velocities. Later approaches provide constraint stabilization

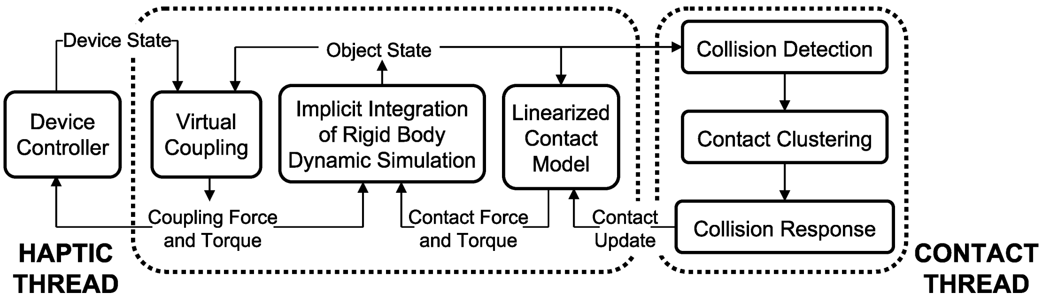

Fig. 1. Multirate architecture. A haptic thread runs at force update rates simulating the dynamics of the virtual tool and computing force feedback, while a contact thread runs asynchronously and updates contact forces.

along with fixed time-stepping [24], [25] but no general guarantees on the passivity of the simulation.

Impulse-based techniques stop the simulation at all collision events and resolve contacts based solely on impulses [26]. Their major drawback is that resting contact leads to multiple micro-collision events. Chang and Colgate [15] integrated passive impulse-based techniques with haptic rendering and stressed the need for other methods to handle resting contact. Recently, Constantinescu *et al.* [27] have proposed the combination of penalty forces with impulsive response. They prove the passivity of multiple impulses applied simultaneously using Newton's restitution law.

Penalty-based techniques apply collision forces based on the amount of object interpenetration [28]. Several researchers have employed penalty-based techniques for haptic rendering, avoiding expensive penetration depth computations by using either local penetration models [1], [12] or precontact penalty forces [11], [13]. Responsive penalty-based forces require the use of high stiffness values, which can compromise the stability of the haptic display in direct rendering approaches or the stability of the simulation of the virtual environment in virtual coupling approaches. Implicit integration is known to provide high stability under larger combinations of mass and stiffness values [29], and it has been used for stable rigid-body simulation with very stiff penalty forces [30], thus avoiding visually perceptible interpenetrations.

III. Algorithm Overview

Our haptic rendering algorithm employs virtual coupling for controlling the impedance displayed to the user. The remainder of the algorithm design decisions are guided by three central goals: to reach a discrete-time passive simulation of the virtual environment, to maximize display transparency, and to maximize the responsiveness of the visual simulation.

We use penalty methods for applying collision response to the virtual tool, as they are especially well suited for fixed time-stepping simulations. Each penalty contact force is continuous-time passive, but the discrete simulation of the virtual environment may not be passive, due to contact discontinuities. We alleviate contact discontinuities by incorporating a contact clustering algorithm that provides spatial smoothing of contact data. We simulate the motion of the virtual tool using implicit integration, which produces a discrete-time passive implementation of the virtual environment up to contact discontinuities. Moreover, implicit integration enhances display transparency by enabling stable simulation of the virtual tool with small mass values, and it also reduces interpenetration of virtual objects by enabling stable simulation with large contact stiffness values. We further alleviate interpenetration problems by applying precontact penalty forces.

We use a linearized contact model for decoupling the rendering algorithm into a *haptic thread* that performs the rigid-body simulation of the virtual tool and a *contact thread* that executes collision detection and response. In this way, collision detection is less a bottleneck for the update rate of the simulation, thereby enabling stiffer coupling impedances. Nevertheless, a frequent update of the linearized contact model is still a requirement with high velocities or geometrically rich objects. Therefore, we incorporate sensation-preserving simplification [9] for performing fast yet perceptually indistinguishable collision detection between complex polygonal models.

The different threads and modules of the rendering algorithm and its implementation are highlighted in Fig. 1. Next, we describe the threads in more detail, and we describe the notation used throughout the paper.

A. Multirate Architecture

The **haptic thread** runs at a high frequency (1 kHz in the experiments described in Section VII), computing rigid-body simulation and force feedback. Each frame, the haptic thread executes the following sequence of operations.

1) Read state of the haptic device at time t_i.
2) Linearize the coupling force and torque at time t_{i-1}.
3) Linearize the contact force and torque at time t_{i-1}.
4) Solve the state of the virtual tool at time t_i, using implicit integration.
5) Compute the coupling force and torque at time t_i.
6) Send the coupling force and torque to the device.

The **contact thread** runs asynchronously at the highest frequency possible given the complexity of the contact scenario, executing the following sequence of operations every update loop.

1) Fetch the state of the virtual tool.
2) Perform collision detection based on sensation-preserving simplification [9].
3) Cluster contacts and compute cluster representatives.
4) For each cluster representative, solve the contact force and torque equations, and compute their Jacobians.

Notation

We use bold-face letters to represent vectors and quaternions and italic upper-case letters to represent matrices. In matrix operations, vectors are in column form. Quaternions may be treated as 4×1 vectors when explicitly indicated. Unless otherwise specified, all magnitudes are expressed in global coordinates of the virtual world. Given a vector $\mathbf{u} = (u_x, u_y, u_z)^T$, $\mathbf{u}^*$ denotes the skew-symmetric matrix used for representing a cross product as a matrix–vector product

$$\mathbf{u}^* = \begin{pmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{pmatrix}. \tag{1}$$

IV. Rigid-Body Dynamics

Here, we formulate the implicit integration for penalty-based dynamic simulation of the virtual tool.

A. *Equations of Rigid-Body Motion*

We formulate the state $\mathbf{y}$ of a rigid body in terms of the position of its center of mass $\mathbf{x}$, a quaternion describing its orientation $\mathbf{q}$, its linear momentum $\mathbf{P}$, and its angular momentum $\mathbf{L}$. With this selection of state variables, the Newton–Euler equations that describe the motion of a rigid body can be written as a function of external forces $\mathbf{F}$ and torques $\mathbf{T}$ by the following ordinary differential equations (ODEs):

$$\dot{\mathbf{y}}(t) = \begin{pmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{q}} \\ \dot{\mathbf{P}} \\ \dot{\mathbf{L}} \end{pmatrix} = \begin{pmatrix} \frac{1}{m}\mathbf{P} \\ \frac{1}{2}\boldsymbol{\omega}_q \mathbf{q} \\ \mathbf{F} \\ \mathbf{T} \end{pmatrix} = \mathbf{f}(\mathbf{y}, t) \tag{2}$$

where m is the mass of the body. The term $\boldsymbol{\omega}_q$ indicates a quaternion with scalar part 0 and vector part the angular velocity $\boldsymbol{\omega}$. Given the mass matrix M of the body, computed in a local frame, and the rotation matrix R from the world frame to the local frame of the body, its angular velocity $\boldsymbol{\omega}$ can be expressed in terms of state variables as

$$\boldsymbol{\omega} = RM^{-1}R^T\mathbf{L}. \tag{3}$$

In many practical applications of 6-DOF haptic rendering (e.g., assembly and disassembly tasks or surgical operations on bones or hard structures), the environment can be considered as static. Following this observation, as many others have done in the past [1], [2], [12], [13], we assume that the only moving object in the simulation is the virtual tool. With this assumption, the state vector $\mathbf{y}$ has 13 variables. The external forces (and similarly for the torques) comprise the weight of the object, penalty-based contact forces $\mathbf{F}_p$, and the virtual coupling force $\mathbf{F}_c$. Friction forces could also be incorporated by using, for example, a local friction model [31].

B. *Implicit Integration*

Implicit discretization of the ODEs using the backward Euler formula yields the following state update:

$$\mathbf{y}_n = \mathbf{y}_{n-1} + \Delta t \dot{\mathbf{y}}_n. \tag{4}$$

Substituting (2) into (4) leads to a nonlinear equation in the state variables $\mathbf{x}$, $\mathbf{q}$, $\mathbf{P}$, and $\mathbf{L}$. A nonlinear solver, such as Newton's method, can be used for finding the exact solution to this system. However, we have decided to trade numerical accuracy for desired speed and linearly approximate (4) using the Taylor expansion of $\mathbf{f}$. This approximation leads to a semi-implicit backward Euler discretization, in which $\partial\mathbf{f}/\partial\mathbf{y}$ is the Jacobian of the equations of rigid-body motion. Rearranging terms, the linear system of equations can be expressed in the form

$$\left(I - \Delta t \frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right)(\mathbf{y}_n - \mathbf{y}_{n-1}) = \Delta t \mathbf{f}_{n-1}. \tag{5}$$

Under the assumption that the virtual tool is the only moving object, $(I - \Delta t \partial\mathbf{f}/\partial\mathbf{y})$ is a 13×13 dense and nonsymmetric matrix. The linear system can be solved by Gaussian elimination. The remainder of this section focuses on the formulation of the Jacobian $\partial\mathbf{f}/\partial\mathbf{y}$.

C. *Jacobian of the Equations of Motion*

The Jacobian of (2) can be expressed as

$$\frac{\partial \mathbf{f}}{\partial \mathbf{y}} = \begin{pmatrix} 0 & 0 & \frac{1}{m}I & 0 \\ 0 & \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{q}} & 0 & \frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{L}} \\ \frac{\partial \mathbf{F}}{\partial \mathbf{x}} & \frac{\partial \mathbf{F}}{\partial \mathbf{q}} & \frac{\partial \mathbf{F}}{\partial \mathbf{P}} & \frac{\partial \mathbf{F}}{\partial \mathbf{L}} \\ \frac{\partial \mathbf{T}}{\partial \mathbf{x}} & \frac{\partial \mathbf{T}}{\partial \mathbf{q}} & \frac{\partial \mathbf{T}}{\partial \mathbf{P}} & \frac{\partial \mathbf{T}}{\partial \mathbf{L}} \end{pmatrix}. \tag{6}$$

The evaluation of $\partial\mathbf{f}/\partial\mathbf{y}$ requires the Jacobians of the equations of external forces (and torques). Sections V and VI deal, respectively, with coupling forces and contact forces.

The expression of the derivative of orientation $\dot{\mathbf{q}}$ is highly nonlinear and leads to two nonzero blocks in the Jacobian, as shown in (6). Given a quaternion $\mathbf{q} = (x, y, z, s)$, the expression of $\dot{\mathbf{q}}$ can be rewritten as a matrix–vector multiplication

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega}_q\mathbf{q} = Q\boldsymbol{\omega}, \qquad Q = \frac{1}{2}\begin{pmatrix} s & z & -y \\ -z & s & x \\ y & -x & s \\ -x & -y & -z \end{pmatrix}. \tag{7}$$

Combining (3) and (7), we obtain the following Jacobians:

$$\frac{\partial \dot{\mathbf{q}}}{\partial \mathbf{L}} = QRM^{-1}R^T \tag{8}$$

$$\frac{\partial \dot{\mathbf{q}}}{\partial q_i} = \frac{\partial Q}{\partial q_i}\boldsymbol{\omega} + Q\frac{\partial \boldsymbol{\omega}}{\partial q_i} \tag{9}$$

$$\frac{\partial \boldsymbol{\omega}}{\partial q_i} = \left(\frac{\partial R}{\partial q_i}M^{-1}R^T + RM^{-1}\frac{\partial R^T}{\partial q_i}\right)\mathbf{L}. \tag{10}$$

Note that $\partial\dot{\mathbf{q}}/\partial\mathbf{q}$ is expressed separately for each of the components q_i of $\mathbf{q}$. Given $\mathbf{q} = (x, y, z, s)$, the partial derivatives of the matrix Q [from (7)] and of the rotation matrix R are as shown in (11) and (12) at the bottom of the page.

V. Virtual Coupling

Here, we describe the equations for coupling force and torque that enable bidirectional interaction with a virtual tool. We also list their Jacobians, which are used in the implicit integration of the motion equations, and we discuss issues associated with device saturation.

A. Coupling Force and Torque

When the virtual tool is *grasped* by the user, the position $\mathbf{x}_h$ and orientation $\mathbf{q}_h$ of the haptic device in the virtual world are recorded as a coupling position $\mathbf{c}$ and coupling orientation $\mathbf{q}_c$ in the local coordinates of the virtual tool

$$\begin{aligned}\mathbf{c} &= R^T(\mathbf{x}_h - \mathbf{x})\\ \mathbf{q}_c &= \mathbf{q}^{-1}\mathbf{q}_h.\end{aligned} \tag{13}$$

During manipulation, the coupling force $\mathbf{F}_c$ is set as a viscoelastic link between the current position of the haptic device and the coupling position. The coupling torque $\mathbf{T}_c$ is composed of the torque induced by the coupling force and a viscoelastic rotational link between the current orientation of the haptic device and the coupling orientation. The rotational link can be expressed in terms of its equivalent axis of rotation $\mathbf{u_c}$. The magnitude of $\mathbf{u_c}$ represents the coupling angle. The coupling force and torque equations are

$$\begin{aligned}\mathbf{F}_c &= k_c(\mathbf{x}_h - \mathbf{x} - R\mathbf{c}) + b_c(\mathbf{v}_h - \mathbf{v} - \boldsymbol{\omega} \times (R\mathbf{c}))\\ \mathbf{T}_c &= (R\mathbf{c}) \times \mathbf{F}_c + k_\theta \mathbf{u_c} + b_\theta(\boldsymbol{\omega}_h - \boldsymbol{\omega})\end{aligned} \tag{14}$$

where k_c and b_c represent linear stiffness and damping, respectively, k_θ and b_θ represent angular stiffness and damping, respectively, and $\mathbf{x}_h$, $\mathbf{v}_h$, and $\boldsymbol{\omega}_h$ represent the position, linear velocity, and angular velocity of the haptic device. The axis of rotation $\mathbf{u_c}$ can be expressed in terms of the rotational coupling deviation $\Delta\mathbf{q}$ and the current orientation $\mathbf{q}$ as

$$\mathbf{u_c} = 2\cos^{-1}(\Delta\mathbf{q}_s)\Delta\mathbf{q}_{xyz} \tag{15}$$

$$\begin{aligned}\Delta\mathbf{q} &= \mathbf{q}_h\mathbf{q}_c^{-1}\mathbf{q}^{-1} = C\mathbf{q}\\ \begin{pmatrix}\Delta\mathbf{q}_{xyz}\\ \Delta\mathbf{q}_s\end{pmatrix} &= \begin{pmatrix}C_{123}\\ C_4\end{pmatrix}\mathbf{q}\end{aligned} \tag{16}$$

where C represents the quaternion product as a matrix–vector multiplication.

B. Jacobians of Coupling Force and Torque Equations

Here, we list the Jacobians of coupling force and torque equations with respect to (w.r.t.) the different state variables. Note that the Jacobians w.r.t. the quaternion are expressed columnwise (i.e., separately for each component q_i of the quaternion)

$$\frac{\partial\mathbf{F}_c}{\partial\mathbf{x}} = -k_c I \tag{17}$$

$$\frac{\partial\mathbf{T}_c}{\partial\mathbf{x}} = -k_c(R\mathbf{c})^* \tag{18}$$

$$\frac{\partial\mathbf{F}_c}{\partial q_i} = -k_c\frac{\partial R}{\partial q_i}\mathbf{c} + b_c(R\mathbf{c})^*\frac{\partial\boldsymbol{\omega}}{\partial q_i} - b_c\boldsymbol{\omega}^*\frac{\partial R}{\partial q_i}\mathbf{c} \tag{19}$$

$$\frac{\partial\mathbf{T}_c}{\partial q_i} = (R\mathbf{c})^*\frac{\partial\mathbf{F}_c}{\partial q_i} - \mathbf{F}_c{}^*\frac{\partial R}{\partial q_i}\mathbf{c} + k_\theta\frac{\partial\mathbf{u_c}}{\partial q_i} - b_\theta\frac{\partial\boldsymbol{\omega}}{\partial q_i} \tag{20}$$

$$\frac{\partial\mathbf{F}_c}{\partial\mathbf{P}} = -\frac{b_c}{m}I \tag{21}$$

$$\frac{\partial\mathbf{T}_c}{\partial\mathbf{P}} = -\frac{b_c}{m}(R\mathbf{c})^* \tag{22}$$

$$\frac{\partial\mathbf{F}_c}{\partial\mathbf{L}} = b_c(R\mathbf{c})^*RM^{-1}R^T \tag{23}$$

$$\frac{\partial\mathbf{T}_c}{\partial\mathbf{L}} = \left(b_c(R\mathbf{c})^*(R\mathbf{c})^* - b_\theta I\right)RM^{-1}R^T. \tag{24}$$

It remains to compute the derivative of the axis of rotation. From (15) and (16), one can obtain the following derivative:

$$\frac{\partial\mathbf{u_c}}{\partial\mathbf{q}} = 2\cos^{-1}\Delta\mathbf{q}_s C_{123} - \frac{2}{\sqrt{1-\Delta\mathbf{q}_s{}^2}}\Delta\mathbf{q}_{xyz}C_4. \tag{25}$$

C. Force Feedback and Device Saturation

After solving the tool state at each frame, we compute coupling force and torque based on (14) using the newly computed tool state. The resulting force and torque values are sent to the device controller as feedback commands.

However, haptic devices present physical limitations that should also be accounted for in the design of virtual coupling. Force (and torque) saturation is one example. When the user pushes against a virtual surface and the device reaches its maximum force value, the user feels no difference as a result of pushing further. The coupling force in the simulation, however, keeps growing, and so does object interpenetration. To avoid this, we model the coupling stiffness k_c as a nonlinear function,

$$\frac{\partial Q}{\partial x} = \frac{1}{2}\begin{pmatrix}0&0&0\\0&0&1\\0&-1&0\\-1&0&0\end{pmatrix},\quad \frac{\partial Q}{\partial y} = \frac{1}{2}\begin{pmatrix}0&0&-1\\0&0&0\\1&0&0\\0&-1&0\end{pmatrix},\quad \frac{\partial Q}{\partial z} = \frac{1}{2}\begin{pmatrix}0&1&0\\-1&0&0\\0&0&0\\0&0&-1\end{pmatrix},\quad \frac{\partial Q}{\partial s} = \frac{1}{2}\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\\0&0&0\end{pmatrix} \tag{11}$$

$$\frac{\partial R}{\partial x} = \begin{pmatrix}2x&y&z\\y&-2x&-s\\z&s&-2x\end{pmatrix},\quad \frac{\partial R}{\partial y} = \begin{pmatrix}-2y&x&s\\x&2y&z\\-s&z&-2y\end{pmatrix},\quad \frac{\partial R}{\partial z} = \begin{pmatrix}-2z&-s&x\\s&-2z&y\\x&y&2z\end{pmatrix},\quad \frac{\partial R}{\partial s} = \begin{pmatrix}2s&-z&y\\z&2s&-x\\-y&x&2s\end{pmatrix} \tag{12}$$

in a way similar to that of Wan and McNeely [2]. We propose a spline stiffness function: 1) for small deviations, under the saturation value, a constant stiffness; 2) a cubic Hermite interpolating function; and 3) for large deviations, zero stiffness. The Jacobians of coupling force and torque equations must be revised, to account for the nonlinearity of the stiffness. From (14), the stiffness-related term of the coupling force $\mathbf{F}_{cx}$ is

$$\mathbf{F}_{cx} = k_c(\mathbf{x}_h - \mathbf{x} - R\mathbf{c}) = k_c \Delta\mathbf{x}. \tag{26}$$

Considering k_c to be a nonlinear function of $\Delta\mathbf{x}$ itself, the Jacobian of the equation of $\mathbf{F}_{cx}$ w.r.t. the tool state $\mathbf{y}$ is expressed as

$$\frac{\partial \mathbf{F}_{cx}}{\partial \mathbf{y}} = \left(k_c I + \Delta\mathbf{x}\frac{\partial k_c}{\partial \Delta\mathbf{x}}\right)\frac{\partial \Delta\mathbf{x}}{\partial \mathbf{y}}. \tag{27}$$

Stability analysis of the nonlinear virtual coupling is desirable, but we have found that it successfully limits object interpenetration under device saturation.

VI. Collision Detection and Response

We begin this section by reviewing the perceptually-based collision detection approach, followed by a description of the contact clustering algorithm. Then, we describe the force and torque equations for collision response, as well as their Jacobians. We conclude the section with the formulation of the linearized contact model.

A. Collision Detection

We perform collision detection using the sensation-preserving simplification algorithm proposed by Otaduy and Lin [9]. As a summary, this algorithm constructs a dual hierarchical representation for each object as part of preprocessing. This representation constitutes a bounding volume hierarchy and a level-of-detail hierarchy simultaneously. At runtime, the collision detection algorithm proceeds by executing a recursive contact query between the hierarchies of two objects. A branch of the query stops if it can be culled away (i.e., the bounding volumes are far apart), or if the current level of detail is perceptually accurate enough for describing contact information. The sensation-preserving simplification algorithm enables the selection of the appropriate geometric resolution at each contact independently.

A contact query returns a set of contacts that sample the regions of the objects that are within a distance tolerance d. Each contact C between the virtual tool and some object in the scene is described by a point $\mathbf{p}$ on the surface of the virtual tool, a point $\mathbf{p}_0$ on the surface of the scene object, the contact normal $\mathbf{n}$ pointing outward from the virtual tool, and the penetration depth δ (which is positive if $\mathbf{p}$ lies inside the scene object, and negative if it lies outside but closer than d).

B. Contact Clustering

A contact query may return multiple contacts to describe each contact region. When using penalty-based collision response, discontinuous motion of the contact points and variability of the number of contacts may jeopardize the passivity of the simulation. We propose a method for grouping contacts based on the K-means clustering technique [32]. Contact clustering limits the number of contacts and, thus, it also limits the total translational stiffness applied to the virtual tool. Moreover, proximity-based clustering provides spatial filtering of contact data for densely sampled objects, and we have found this useful for alleviating discontinuities.

Given a set of n contacts $\{C_0, C_1, \ldots, C_{n-1}\}$, we define K clusters $\{S_0, S_1, \ldots, S_{K-1}\}$, and compute a representative contact for each cluster. Penalty-based contact forces are computed at the representative contacts. If the number of input contacts is $n < K$, we only create n clusters.

The clusters are defined implicitly by storing an additional parameter along with each contact C: the cluster it belongs to, S. Then, a contact C is defined as a tuple $(\mathbf{p}, \mathbf{p}_0, \mathbf{n}, \delta, S)$. In the description of the clustering algorithm, we reference each parameter of a contact as $C.\text{parameter}$ (e.g., $C.\mathbf{p}$). Similarly, we define a cluster S as a tuple $(\mathbf{p}, \mathbf{p}_0, \mathbf{n}, \delta)$, where $\mathbf{p}$, $\mathbf{p}_0$, $\mathbf{n}$, and δ are the contact parameters of the cluster representative.

We formulate a cost function f for the K-means clustering problem, based on the Euclidean distance between each contact point $C.\mathbf{p}$ and the representative of the cluster it belongs to, $C.S.\mathbf{p}$, weighted by the penetration depth of the contact $C.\delta$. We have found this strategy beneficial for increasing the smoothness of penalty-based collision response. Specifically, the cost function f is written as

$$f = \frac{\sum_i^{n-1}\left((C_i.\delta + d)\|C_i.\mathbf{p} - C_i.S.\mathbf{p}\|^2\right)}{\sum_i^{n-1}(C_i.\delta + d)}. \tag{28}$$

This cost function is minimized when the cluster representatives are located at the centroids of the clusters. This property is exploited by Lloyd's method [33], which is a greedy algorithm that solves the K-means clustering problem by interleaving one step of centroid computation with one step of reclustering until the clusters converge. We have adapted Lloyd's method to compute contact clusters, because the clustering is expected to converge rapidly by exploiting temporal coherence and initializing cluster centroids at the positions of representative contacts from the previous frame. At every iteration of Lloyd's method, we reassign each contact to its closest representative, and we recompute the position of the representative of each cluster as the centroid of all the contact points in the cluster, weighted by their penetration depth. The expression for the position of each representative is

$$S.\mathbf{p} = \frac{\sum_{i,\, C_i.S=S}((C_i.\delta + d)C_i.\mathbf{p})}{\sum_{i,\, C_i.S=S}(C_i.\delta + d)}. \tag{29}$$

Contacts must be clustered at every execution of the contact thread. The clustering information from the previous frame can be used to initialize the iterative process of Lloyd's method. The first step of the initialization is to determine the number of output clusters m. Then, if m is smaller than the number of input clusters l, we drop the input clusters with smallest penetration depth. Next, we initialize the positions of the representatives of $\min(m, l)$ output clusters at the contact points that are closest to the representatives of the remaining input clusters. If m is larger than the number of input clusters, we must still initialize the representatives of $m - l$ output clusters. We place these representatives at the contact points that are furthest from the output cluster representatives that are already initialized. Initializing the representatives at contact points ensures that every cluster contains at least one contact.

Once the clusters converge, we compute the remaining parameters of the representative contact for each cluster (i.e., $\mathbf{p}_0$, δ, and $\mathbf{n}$), based on the following expressions:

$$S.\delta = \frac{\sum_{i,\, C_i.S=S} ((C_i.\delta + d)C_i.\delta)}{\sum_{i,\, C_i.S=S} (C_i.\delta + d)} \tag{30}$$

$$S.\mathbf{n} = \frac{\hat{\mathbf{n}}}{\|\hat{\mathbf{n}}\|}$$
$$\hat{\mathbf{n}} = \frac{\sum_{i,\, C_i.S=S} ((C_i.\delta + d)C_i.\mathbf{n})}{\sum_{i,\, C_i.S=S} (C_i.\delta + d)} \tag{31}$$

$$S.\mathbf{p}_0 = S.\mathbf{p} - S.\delta(S.\mathbf{n}). \tag{32}$$

Algorithm VI.1 shows the pseudocode for contact clustering based on Lloyd's method.

ALGORITHM VI.1: Contact Clustering Based on Lloyd's Method

```
$\{S\} \leftarrow$ CLUSTER_CONTACTS($\{C\}, \{S'\}$)

Input: The set of new contacts $\{C_0, C_1, \ldots C_{n-1}\}$ and the set of old clusters $\{S'_0, S'_1, \ldots, S'_{l-1}\}$, assuming that the old clusters are ordered according to decreasing $\delta$.
Output: The set of new clusters $\{S_0, S_1, \ldots S_{m-1}\}$.

Initialize $\{S\} \leftarrow$ INIT_CLUSTERS($\{C\}, \{S'\}$)
  repeat
    for each contact $C_i$ do
      Assign cluster $C_i.S \leftarrow \min_{S_j \in \{S\}} ||S_j.\mathbf{p} - C_i.\mathbf{p}||$
    for each cluster $S_i$ do
      Compute representative $S_i.\mathbf{p}$ according to (29)
  until the clusters converge
  for each cluster $S_i$ do
    Compute parameters of the representative ($S_i.\delta$, $S_i.\mathbf{n}$, and $S_i.\mathbf{p}_0$) according to (30)–(32)
```

```
  $\{S\} \leftarrow$ INIT_CLUSTERS($\{C\}, \{S'\}$)

Input: A set of contacts $\{C_0, C_1, \ldots, C_{n-1}\}$ and the set of old clusters $\{S'_0, S'_1, \ldots, S'_{l-1}\}$.
Output: A new set of clusters $\{S_0, S_1, \ldots S_{m-1}\}$ with initial representative positions.
  $m = \min(K, n)$
  if $l > m$
    Remove clusters with small $\delta$ $\{S'_m, \ldots S'_{l-1}\}$ from $\{S'\}$
  for each new cluster $S_i$ s.t. $i < \min(l, m)$ do
    Find closest pair
      $(C_j, S'_k) \leftarrow \min_{C_j \in \{C\}} \min_{S'_k \in \{S\}} ||C_j.\mathbf{p} - S'_k.\mathbf{p}||$
    Remove $C_j$ from $\{C\}$
    Remove $S'_k$ from $\{S'\}$
    Assign representative $S_i.\mathbf{p} \leftarrow C_j.\mathbf{p}$
    Add $S_i$ to $\{S\}$
  for each new cluster $S_i$ s.t. $l \le i < m$ do
    Find furthest contact
      $C_j \leftarrow \max_{C_j \in \{C\}} \min_{S_k \in \{S\}} ||C_j.\mathbf{p} - S_k.\mathbf{p}||$
    Remove $C_j$ from $\{C\}$
    Assign representative $S_i.\mathbf{p} \leftarrow C_j.\mathbf{p}$
    Add $S_i$ to $\{S\}$
```

C. Penalty-Based Collision Response

After contact clustering, the contact normal $\mathbf{n}$ is a representative value that does not capture exact information about surface features, therefore we have opted to model each contact as a planar constraint. The constraint is represented by the plane with normal $\mathbf{n}$ and passing through $\mathbf{p}_0$. Note that it is also convenient to represent $\mathbf{p}$ based on its coordinates in the local frame of the virtual tool $\mathbf{r}$. We compute viscoelastic penalty-based force $\mathbf{F}_p$ and torque $\mathbf{T}_p$ as

$$\begin{aligned}\mathbf{F}_p &= -\,kN(\mathbf{x} + R\mathbf{r} - \mathbf{p}_0) - k\,d\,\mathbf{n} - bN(\mathbf{v} + \boldsymbol{\omega} \times (R\mathbf{r}))\\ \mathbf{T}_p &= (R\mathbf{r}) \times \mathbf{F}_p.\end{aligned} \tag{33}$$

N is a matrix that projects a vector onto the normal of the constraint plane, and it is computed as $\mathbf{n}\,\mathbf{n}^T$.

D. Jacobians of Collision Response Equations

Here, we list the Jacobians of penalty-based force and torque equations w.r.t. the different state variables. Note that the Jacobians w.r.t. the quaternion are expressed columnwise, and the contact normal is considered to be constant during one frame of the simulation

$$\frac{\partial \mathbf{F}_p}{\partial \mathbf{x}} = -\,kN \tag{34}$$
$$\frac{\partial \mathbf{T}_p}{\partial \mathbf{x}} = (R\mathbf{r})^* \frac{\partial \mathbf{F}_p}{\partial \mathbf{x}} \tag{35}$$
$$\frac{\partial \mathbf{F}_p}{\partial q_i} = -\,kN\frac{\partial R}{\partial q_i}\mathbf{r} - bN\boldsymbol{\omega}^*\frac{\partial R}{\partial q_i}\mathbf{r} + bN(R\mathbf{r})^*\frac{\partial \boldsymbol{\omega}}{\partial q_i} \tag{36}$$
$$\frac{\partial \mathbf{T}_p}{\partial q_i} = (R\mathbf{r})^*\frac{\partial \mathbf{F}_p}{\partial q_i} - \mathbf{F}_p{}^*\frac{\partial R}{\partial q_i}\mathbf{r} \tag{37}$$

$$\frac{\partial \mathbf{F}_p}{\partial \mathbf{P}} = -\frac{b}{m}N \tag{38}$$
$$\frac{\partial \mathbf{T}_p}{\partial \mathbf{P}} = (R\mathbf{r})^* \frac{\partial \mathbf{F}_p}{\partial \mathbf{P}} \tag{39}$$
$$\frac{\partial \mathbf{F}_p}{\partial \mathbf{L}} = bN(R\mathbf{r})^* RM^{-1}R^T \tag{40}$$
$$\frac{\partial \mathbf{T}_p}{\partial \mathbf{L}} = (R\mathbf{r})^* \frac{\partial \mathbf{F}_p}{\partial \mathbf{L}}. \tag{41}$$

E. Linearized Contact Model

In complex contact configurations, collision detection may easily run at rates notably slower than the update of rigid-body dynamics, even with sensation-preserving simplification [9]. In such cases, linear approximations of the contact force equations increase the accuracy of the derivatives of state variables and thereby the stability of implicit integration. Assuming that the contact thread performed the last update of contact force (and similarly for the torque) at time t, the contact force $\mathbf{F}_p$ at time $t + \Delta t$ can be linearly approximated using its Taylor expansion as

$$\mathbf{F}_p(t+\Delta t) = \mathbf{F}_p(t) + \frac{\partial \mathbf{F}_p}{\partial \mathbf{y}}(t)\left(\mathbf{y}(t+\Delta t) - \mathbf{y}(t)\right). \tag{42}$$

Note that penalty-based contact forces depend solely on the state of the virtual tool, therefore $\partial\mathbf{F}_p/\partial t = 0$ and $\partial\mathbf{T}_p/\partial t = 0$. The Jacobians of contact force and torque equations w.r.t. the tool state $\mathbf{y}$ must also be computed for the semi-implicit formulation of backward Euler integration. Therefore, the computation of the linearized contact model has little additional cost. The linearized contact model can potentially be recomputed at the rate of the haptic thread, but we found little performance improvement by doing this. A probable reason is that the accuracy of the linearized contact model depends mostly on the accuracy of the contact points and normals, and these data are only updated by the contact thread.

VII. Experiments and Results

A. Implementation Details

The experiments have been performed using a dual Pentium-4 2.4-GHz processor PC with 2.0 GB of memory and an NVidia GeForce FX5950 graphics card, and Windows2000 OS. We have used a 6-DOF *Phantom*™ impedance-type haptic device, but our formulation is also applicable to admittance-type haptic devices, following Adams and Hannaford's framework [14]. The haptic thread is executed at a constant frequency of 1 kHz, and it employs utilities of GHOST-SDK, the software API of the *Phantom* haptic device, to communicate with the device controller. The contact thread is executed asynchronously and is assigned a lower scheduling priority.

B. Free-Space Motion

We have designed an experiment to evaluate the transparency of the rendering algorithm during free-space motion with virtual coupling. In the experiment, the haptic device commands the motion of a 20-cm-long spoon (see Fig. 2). The spoon is moved freely, without touching other objects. Our goal was to maximize transparency by minimizing the mass of the virtual tool (i.e., the spoon) and thereby the feedback forces. A thin object, such as a spoon, is particularly challenging for the stability of numerical integration due to its low inertia around its longitudinal axis.

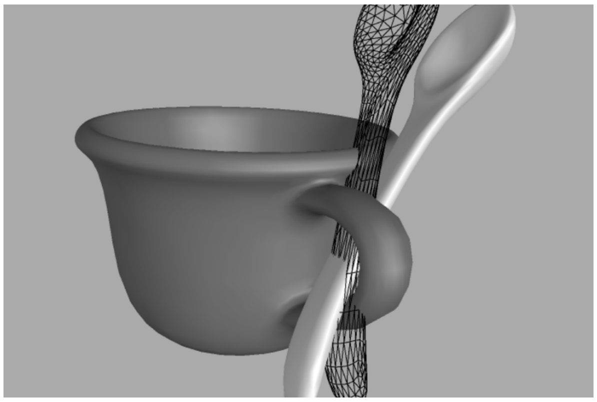
Fig. 2. Manipulation of a spoon in contact with a cup using virtual coupling. As the spoon is constrained inside the handle of the cup, the contact force and torque are perceived through a virtual coupling. A wireframe image of the spoon represents the actual configuration of the haptic device.

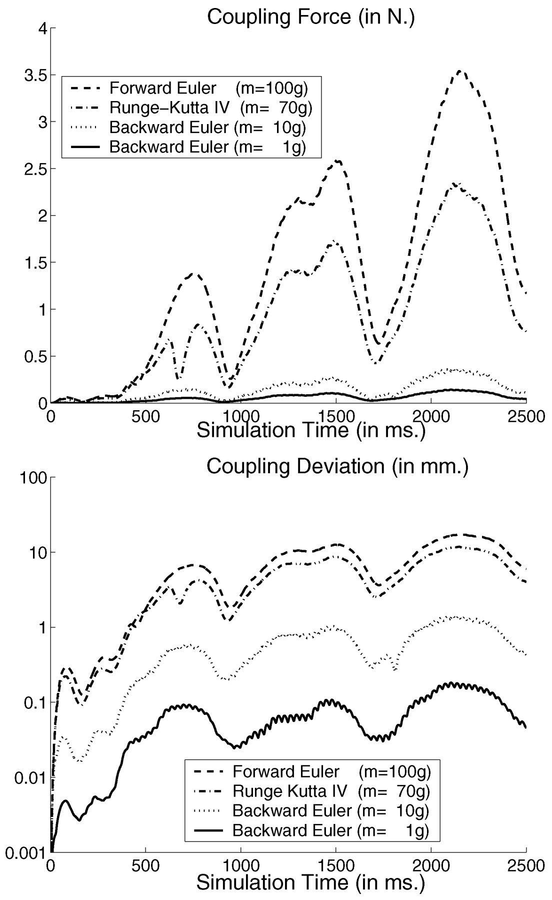

Fig. 3. Coupling deviation and force during free-space motion. Comparisons using different numerical integration methods, and varying the mass of the virtual tool. Top: log plot of the coupling deviation. Bottom: coupling force.

Fig. 3 reflects the coupling deviation $\|\mathbf{x}_h - \mathbf{x}_c\|$ and the absolute value of coupling force $\|\mathbf{F}_c\|$ during 2.5 s of simulation. We have collected the values of coupling deviation and force using different numerical integration methods (i.e., forward

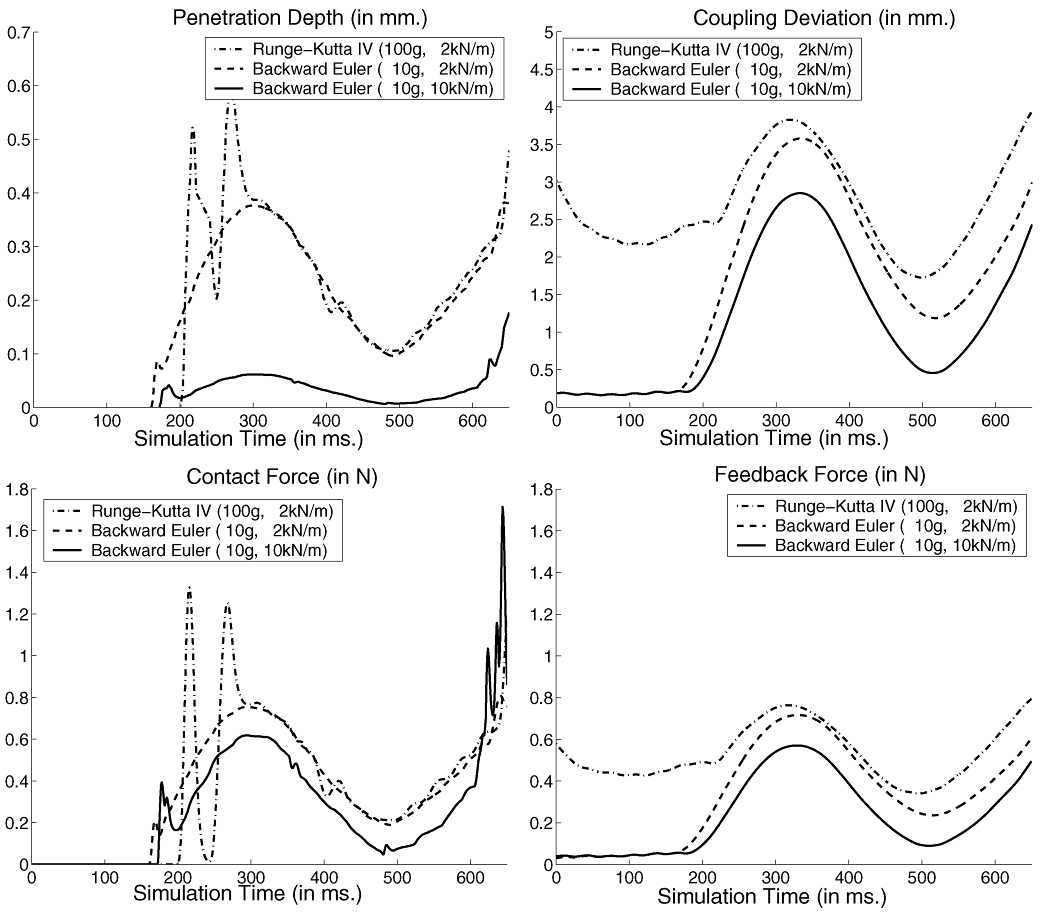

Fig. 4. Forces and positions during contact. Comparison of maximum local penetration depth (top left), coupling deviation (top right), contact force (bottom left), and feedback or coupling force (bottom right) using different numerical integration methods and contact stiffness values.

Euler, Runge–Kutta IV, and backward Euler), for the same trajectory of the haptic device. This trajectory was recorded using the suggested backward Euler as the integration method, while rendering interactively the coupling forces to the user. With the implicit backward Euler as the integration method, coupling stiffness $k_c = 200$ N/m, and $k_\theta = 0.6$ Nm/rad, the simulation is stable with a mass as small as 1 g. However, with Runge–Kutta IV and forward Euler, the simulation is stable only with masses larger than 70 and 100 g, respectively. As can be deduced from Fig. 3, smaller stable mass values lead to more transparent display, in the form of smaller coupling deviations and forces.

C. Experiments During Contact

A scenario with relatively simple models (i.e., the cup and the spoon depicted in Fig. 2) has been used to compare the effects of different integration methods and contact stiffness values on the stability and transparency of the system in contact situations. Using our haptic rendering algorithm, we have displayed the interaction force and torque while the user manipulated the virtual spoon (i.e., 1344 triangles and 20-cm long) in contact with the virtual cup (i.e., 4000 triangles and 8-cm radius). We have used the following parameter settings: backward Euler, $m = 10$ g, $k = 2$ kN/m, and $k_c = 200$ N/m. The trajectory of the haptic device was then recorded and played with different settings as well: 1) Runge–Kutta IV, $m = 100$ g, $k = 2$ kN/m, and $k_c = 200$ N/m; 2) backward Euler, $m = 10$ g, $k = 10$ kN/m, and $k_c = 200$ N/m. Fig. 4 shows graphs of maximum local penetration depth (top left), coupling deviation (top right), contact force (bottom left), and feedback or coupling force (bottom right) during 650 ms of the simulation with the different settings.

As can be inferred from the graph of penetration depth in Fig. 4, the spoon moved in free space for a period of more than 100 ms and then started penetrating the surface of the cup. The spoon remained in contact with the cup (penetrating slightly) during the remainder of the simulation.

Numerical integration of the simulation of the spoon with the Runge–Kutta IV method is stable for values of the mass larger than 70 g, as concluded from the experiments in free-space motion. This requirement affects the transparency during contact state as well. As reflected in the bottom right graph of Fig. 4, with a mass of 100 g, the magnitude of feedback force during free-space motion and contact situations is very similar. This similarity degrades the kinesthetic perception of contact. Implicit integration, however, is stable for small values of the mass, and this produces a clear distinction in the magnitude of feedback force between free-space motion and contact state.

High contact stiffness minimizes the amount of interpenetration between the spoon and the cup. As shown in the top left graph of Fig. 4, the maximum penetration during the interval of study was smaller than 0.6 mm with a contact stiffness of 2 kN/m. The results in Fig. 4 show that, by combining stiff penalty-based collision response with implicit integration, we

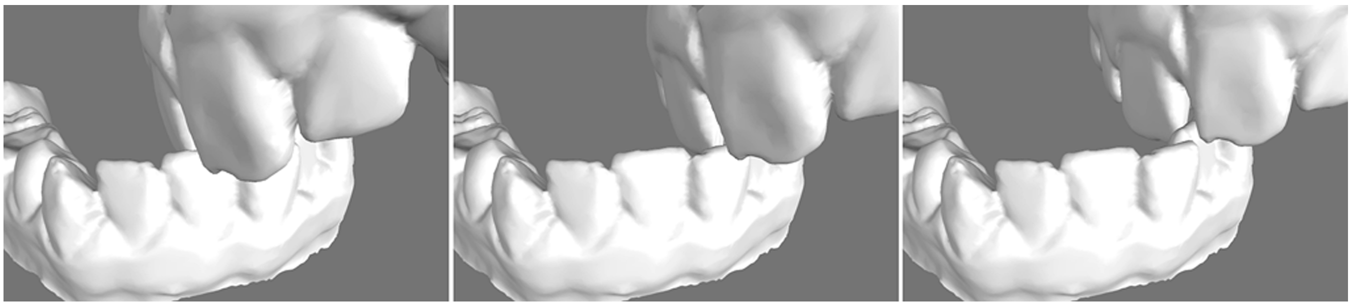

Fig. 5. Dexterous interaction of virtual jaws. Three snapshots of an upper jaw (47 339 triangles) being moved over a lower jaw (40 180 triangles), with intricate teeth interaction.

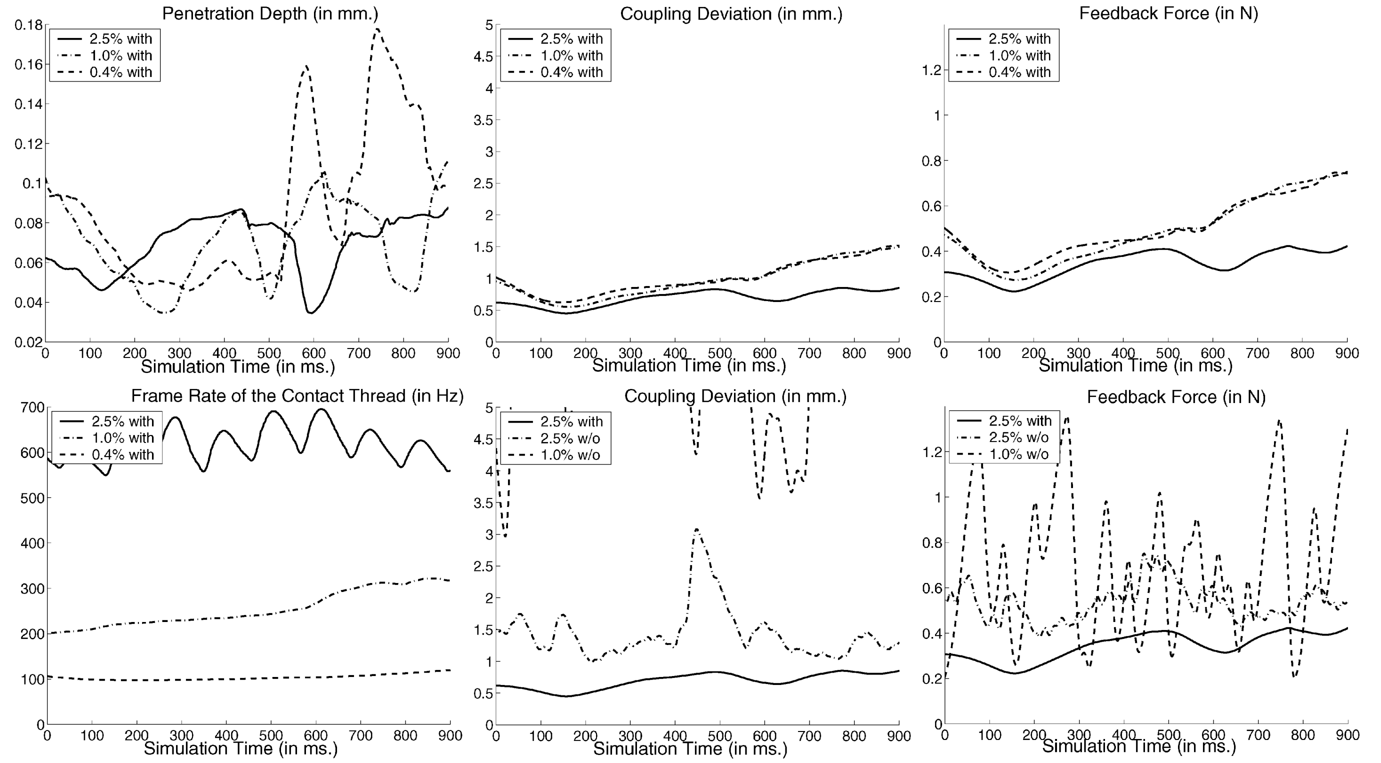

Fig. 6. Effects of the linearized contact model. Comparison of maximum local penetration depth (top left), frame rate of the contact thread (bottom left), coupling deviation (center), and feedback or coupling force (right) using different error tolerances for sensation-preserving simplification, with and without (w/o) linearized contact model.

have been able to display stable forces with small visual interpenetrations, which enhance the perception of hard contact.

However, the display is susceptible to instability problems with high contact stiffness. Contact clustering alleviates the discontinuities of contact-point positions, but (smaller) discontinuities are still present, which can jeopardize the passivity of the simulation. The left graphs of Fig. 4 show unstable behavior with Runge–Kutta IV and $k = 2$ kN/m, and with backward Euler and $k = 10$ kN/m. Out of the interval of study, the oscillations with these settings became more serious, and were also transmitted to the coupling force. Nevertheless, with Backward Euler and $k = 2$ kN/m, the simulation and the display remained stable.

D. Experiments With Complex Models

A scenario with two complex virtual jaws (see Fig. 5) has been used to test the effectiveness of contact clustering, the linearized contact model and the stability and transparency of our haptic rendering algorithm on complex polygonal models. The model of the lower jaw is composed of 40 180 triangles, while the upper jaw consists of 47 339 triangles.

We recorded a trajectory of the upper jaw while rendering the interaction with the lower jaw and using sensation-preserving simplification [9] with an error threshold of 2.5% of the radius of the jaws. Then, we played this same trajectory with smaller error thresholds of 1% and 0.4%, thereby increasing the cost of collision detection and decreasing the update rate of the contact thread. We ran the experiments with and without the use of the linearized contact model. In the experiment without linearized contact model and with an error threshold of 0.4%, the simulation soon became unstable and the position of the upper jaw diverged to infinity. For clarity of the graphs, we have not included the data for this part of the experiment.

Fig. 6 shows graphs of maximum local penetration depth (top left), frame rate of the contact thread (bottom left), coupling deviation (center), and feedback or coupling force (right) during 900 ms of simulation. The models of both jaws can be bounded by spheres of 6-cm radius. We scaled the workspace of the device by a factor of 0.4, therefore, the forces plotted in the graphs were scaled by a factor of 2.5 before being displayed back to the user. All experiments were executed using backward Euler implicit integration, a mass $m = 10$ g for the upper jaw, coupling stiffness $k_c = 500$ N/m, and contact stiffness $k = 5$ kN/m.

The plots demonstrate that with our linearized contact model and an error threshold of 2.5%, the rendering was stable and highly transparent. For example, the maximum local penetration depth never exceeded 0.1 mm with a contact stiffness as high as 5 kN/m. With the linearized contact model but reducing the error threshold, interpenetrations were larger, but the rendering remained stable. With an error threshold of 0.4%, the update rate of the contact thread dropped to 100 Hz at times. Even in such a challenging situation, the high update rate of the linearized contact forces kept the display stable.

On the other hand, without the linearized contact model, the performance degraded rapidly. Even with an error threshold of 2.5%, which kept the update rate of the contact thread over 500 Hz, the feedback force became clearly unstable. The comparison of simulation data with and without the linearized contact model clearly indicates the influence of the linearized contact model on the stability of the system when the update rate of the contact thread decays. This observation demonstrates that the linearized contact model is a key factor for successful 6-DOF haptic rendering of complex models.

In the benchmark of the interacting jaws, we executed the contact clustering algorithm described in Section VI-B with a target number of clusters $K = 5$. In the experiment with the linearized contact model and an error threshold of 0.4%, the number of contacts before clustering is at times as high as 50. This would imply that the total contact stiffness applied to the upper jaw could grow up to 250 kN/m (which is 50 times higher than the nominal stiffness), probably inducing unstable behavior. However, the contact clustering algorithm clamps the number of contacts employed in collision response to five, thus limiting the total translational contact stiffness applied to the upper jaw to 25 kN/m and enhancing stability, as shown in Fig. 6.

VIII. Conclusion

In this paper, we have presented a 6-DOF haptic rendering algorithm that enables stable and transparent interaction between rigid models with tens of thousands of triangles. We simulate the motion of the virtual tool using implicit integration of rigid-body dynamics and penalty-based collision response, and we render interaction forces through a virtual coupling. We have incorporated a fast, perceptually-based collision detection algorithm [9] and a linearized contact model for ensuring a very high update rate of the simulation and the feedback forces. Moreover, a novel contact clustering technique reduces instability problems associated with penalty-based collision response. The complete integration of the perceptually based collision detection algorithm in the rendering algorithm is extensively described in [34].

Our rendering algorithm exploits virtual coupling for simplifying the design of a passive display, but it enhances transparency by using implicit integration and by maximizing the update rate of the simulation of the virtual tool. Other existing techniques could also be considered for the purpose of enhancing transparency. On the one hand, approximate incremental collision detection algorithms [13] can offer fast force updates, in a way similar to the linearized contact model. The main benefit of the linearized contact model is a strictly constant and very small cost. On the other hand, quasi-static approximation of the motion of the virtual tool [2] appears as an alternative for massless manipulation. However, this approach disables the possibility of rendering viscous and inertial effects, and further passivity analysis is required.

Despite the benefits of contact clustering, the use of penalty-based collision response may introduce rendering instabilities. We have found that the rendering was stable with rather high contact stiffness values, which may well be because discrete-time passivity of the simulation along with a properly tuned virtual coupling is a sufficient condition for stability of the display and not a necessary condition. However, the lack of stability guarantees suggests the need for better collision resolution methods. To this regard, recent research in the passivity of sequentially activated force models [20], as well as constraint-based simulation methods with fixed time-stepping [24], [25] seems highly promising. Even in the case of constraint-based simulation, many of the modules of the rendering algorithm presented in this paper (i.e., perceptually based collision detection, contact clustering, linearized contact models, and implicit integration) would be beneficial for enhancing the transparency of the haptic display.

Our rendering algorithm presents still many limitations in terms of the description of the virtual environment. Friction forces can easily be added using localized friction models [31]. Also, the algorithm can be extended to dynamic environments, but the cost of the simulation will grow considerably for complex scenes. Finally, other collision detection and simulation techniques can be investigated for handling deformable bodies, textured surfaces, and other types of model representations.

To conclude, our work can benefit from studies of human factors, since the stability and transparency of the rendering algorithm can be evaluated from a perceptual perspective. Also, it can also benefit from its integration with practical applications, such as training simulators for endoscopic surgery, to help us identify future research needs.

Acknowledgment

The authors would like to thank E. Colgate, the UNC Gamma Group, and the anonymous reviewers for their feedback on the earlier drafts of this paper.

References

[1] W. McNeely, K. Puterbaugh, and J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling," in *Proc. ACM SIGGRAPH*, 1999, pp. 401–408.

[2] M. Wan and W. A. McNeely, "Quasi-static approximation for 6 degrees-of-freedom haptic rendering," in *Proc. IEEE Visualization Conf.*, 2003, pp. 257–262.

[3] C. Edmond, D. Heskamp, D. Sluis, D. Stredney, G. Wiet, R. Yagel, S. Weghorst, P. Oppenheimer, J. Miller, M. Levin, and L. Rosenberg, "ENT endoscopic surgical simulator," in *Proc. Medicine Meets VR*, 1997, pp. 518–528.

[4] V. Hayward, P. Gregorio, O. Astley, S. Greenish, and M. Doyon, "Freedom-7: A high fidelity seven axis haptic device with applications to surgical training," in *Experimental Robotics*. Berlin, Germany: Springer-Verlag, 1998, vol. 232, Lecture Notes in Control and Information Sciences, pp. 445–456.

[5] J. E. Colgate and G. G. Schenkel, "Passivity of a class of sampled-data systems: Application to haptic interfaces," in *Proc. Amer. Control Conf.*, 1994, pp. 3236–3240.
[6] F. P. Brooks, Jr., M. Ouh-Young, J. J. Batter, and P. J. Kilpatrick, F. Baskett, Ed., "Project GROPE—Haptic displays for scientific visualization," in *Proc. Comput. Graph. (SIGGRAPH)*, Aug. 1990, vol. 24, pp. 177–185.
[7] J. E. Colgate, M. C. Stanley, and J. M. Brown, "Issues in the haptic display of tool use," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1995, pp. 140–145.
[8] Y. Adachi, T. Kumano, and K. Ogino, "Intermediate representation for stiff virtual objects," in *Proc. Virtual Reality Annu. Int. Symp.*, 1995, pp. 203–210.
[9] M. A. Otaduy and M. C. Lin, "Sensation preserving simplification for haptic rendering," in *Proc. ACM SIGGRAPH*, 2003, pp. 543–553.
[10] S. E. Salcudean and T. D. Vlaar, "On the emulation of stiff walls and static friction with a magnetically levitated input/output device," in *Proc. ASME Haptic Interfaces for Virtual Environ. Teleoperator Syst.*, 1994, pp. 303–310.
[11] A. Gregory, A. Mascarenhas, S. Ehmann, M. C. Lin, and D. Manocha, "6-DOF haptic display of polygonal models," in *Proc. IEEE Visualization Conf.*, 2000, pp. 139–146.
[12] Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha, "Six-degree-of-freedom haptic rendering using incremental and localized computations," *Presence*, vol. 12, no. 3, pp. 277–295, 2003.
[13] D. E. Johnson and P. Willemsen, "Accelerated haptic rendering of polygonal models through local descent," in *Proc. Haptics Symp.*, 2004, pp. 18–23.
[14] R. J. Adams and B. Hannaford, "A two-port framework for the design of unconditionally stable haptic interfaces," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1998, pp. 1254–1259.
[15] B. Chang and J. E. Colgate, "Real-time impulse-based simulation of rigid body systems for haptic display," in *Proc. ASME Dyn. Syst. Control Div.*, 1997, pp. 145–152.
[16] P. J. Berkelman, "Tool-based haptic interaction with dynamic physical simulations using Lorentz magnetic levitation," Ph.D. dissertation, Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, 1999.
[17] D. Ruspini and O. Khatib, "A framework for multi-contact multi-body dynamic simulation and haptic display," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2000, pp. 1322–1327.
[18] B. E. Miller, J. E. Colgate, and R. A. Freeman, "Guaranteed stability of haptic systems with nonlinear virtual environments," *IEEE Trans. Robot. Autom.*, vol. 16, no. 6, pp. 712–719, Dec. 2000.
[19] B. Hannaford, J.-H. Ryu, and Y. S. Kim, , M. L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme, Eds., "Stable control of haptics," in *Touch in Virtual Environments*. Upper Saddle River, NJ: Prentice-Hall, 2002, ch. 3, pp. 47–70.
[20] M. Mahvash and V. Hayward, "High-fidelity passive force-reflecting virtual environments," *IEEE Trans. Robot.*, vol. 21, no. 1, pp. 38–46, Feb. 2005.
[21] M. A. Otaduy, N. Jain, A. Sud, and M. C. Lin, "Haptic display of interaction between textured models," in *Proc. IEEE Visualization Conf.*, 2004, pp. 297–304.
[22] Q. Luo and J. Xiao, "Physically accurate haptic rendering with dynamic effects," *IEEE Comput. Graph. Appl.*, vol. 24, no. 6, pp. 60–69, 2004.
[23] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and Coulomb friction," *Int. J. Numer. Methods Eng.*, vol. 39, no. 14, pp. 2673–2691, 1996.
[24] M. B. Cline and D. K. Pai, "Post-stabilization for rigid body simulation with contact and constraints," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2003, pp. 3774–3551.
[25] M. Anitescu and G. D. Hart, "A constraint-based time-stepping approach for rigid multibody dynamics with joints, contact and friction," *Int. J. Numer. Methods Eng.*, vol. 60, no. 14, pp. 2335–2371, 2004.
[26] B. V. Mirtich, "Impulse-based dynamic simulation of rigid body systems," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, 1996.
[27] D. Constantinescu, S. E. Salcudean, and E. A. Croft, "Haptic rendering of rigid contacts using impulsive and penalty forces," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 309–323, Jun. 2005.
[28] M. Moore and J. Wilhelms, "Collision detection and response for computer animation," *Comput. Graph.*, vol. 22, no. 4, pp. 289–298, 1988.
[29] D. Baraff and A. Witkin, "Large steps in cloth simulation," in *Proc. ACM SIGGRAPH*, 1998, pp. 43–54.
[30] D. Wu, "Penalty methods for contact resolution," presented at the Game Developers Conf. 2000.
[31] V. Hayward and B. Armstrong, "A new computational model of friction applied to haptic rendering," *Exp. Robot.*, vol. VI, pp. 404–412, 2000.
[32] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
[33] S. P. Lloyd, Least squares quantization in PCM's Bell Telephone Labs. , 1957, Tech. Memo.
[34] M. A. Otaduy, "6-DOF haptic rendering using contact levels of detail and haptic textures," Ph.D. dissertation, Dept. Comput. Sci., Univ. North Carolina at Chapel Hill, Chapel Hill, NC, 2004.

Miguel A. Otaduy received the B.S. degree in electrical engineering from Mondragon Unibertsitatea, Mondragon, Spain, in 2000, and the M.S. and Ph.D. degrees in computer science from the University of North Carolina (UNC), Chapel Hill, in 2003 and 2004, respectively. His dissertation was in the field of haptic rendering.

He is currently a Post-Doctoral Research Associate with the Computer Graphics Laboratory, ETH-Zurich, Zurich, Switzerland. Between 1995 and 2000, he was a Research Assistant with Ikerlan Research Laboratory, and between 2000 and 2004, he was a Research Assistant with the Gamma Group, UNC. In the summer of 2003, he was with Immersion Medical. His research areas include physically-based simulation, haptic rendering, collision detection, medical applications, and geometric algorithms. He has taught tutorials on haptic rendering in the ACM SIGGRAPH and Eurographics international conferences, and has served on the program committees of Pacific Graphics, Computer Graphics International, and the Eurographics Symposium on Virtual Environments.

Dr. Otaduy was the recipient of fellowships from the Government of the Basque Country and the UNC Computer Science Alumni.

Ming C. Lin (S'90–M'90) received the B.S., M.S., and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 1988, 1991, and 1993, respectively.

She is currently a Full Professor with the Department of Computer Science, University of North Carolina (UNC), Chapel Hill. Prior to joining UNC, she was an Assistant Professor with the Computer Science Department at both the Naval Postgraduate School, Monterey, CA, and North Carolina A&T State University, Greensboro, and a Program Manager with the U.S. Army Research Office. Her research interests include physically-based modeling, haptics, robotics, real-time 3-D graphics for virtual environments, geometric computing, and distributed interactive simulation. She has authored more than 140 refereed publications in these areas. She has served as a program committee member for many leading conferences on virtual reality, computer graphics, robotics, and computational geometry. She was the general chair and/or program chair of several conferences, including the ACM Workshop on Applied Computational Geometry 1996, ACM Symposium on Solid Modeling and Applications 1999, Workshop on Intelligent Human Augementation and Virtual Environments 2002, ACM SIGGRAPH/EG Symposium on Computer Animation 2003, ACM Workshop on General Purpose Computing on Graphics Processors 2004, Eurographics 2005, Computer Animation and Social Agents 2005, and Eurographics Symposium on Virtual Environments 2006. She also serves on the Steering Committee of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, the Advisory Board of IEEE World Haptics Conference, and the National Science Foundation (NSF) Information Technology Research Committee of Visitors. She has served as an Associate Editor or Guest Editor for several journals and magazines, including the *International Journal on Computational Geometry and Applications* and *ACM Computing Reviews in Computer Graphics*. She also coedited the book *Applied Computation Geometry* (New York: Springer, 1996).

Dr. Lin was the recipient of several honors and awards, including the NSF Young Faculty Career Award in 1995, the Honda Research Initiation Award in 1997, the UNC/IBM Junior Faculty Development Award in 1999, the UNC Hettleman Award for Scholarly Achievements in 2002, and Best Paper Awards at the Army Science Conference 1996, Eurographics 1999, Eurographics 2002, and ACM Symposium in Solid Modeling and Applications 2003, and IEEE Virtual Reality Conference 2005. She has served as an Associate Editor or Guest Editor of the IEEE TRANSACTIONS ON COMPUTER GRAPHICS AND VISUALIZATION and the *IEEE Computer Graphics and Applications* magazine.