Computers & Graphics 34 (2010) 689–697

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Computer Graphics in Spain: a Selection of Papers from CEIG 2009

Haptic rendering of objects with rigid and deformable parts

Carlos Garre *, Miguel A. Otaduy

GMRV - Universidad Rey Juan Carlos, Madrid, Spain

ARTICLE INFO

Keywords:
Haptic rendering
Force linearization
Hand interaction
Deformable tool

ABSTRACT

In many haptic applications, the user interacts with the virtual environment through a rigid tool. Tool-based interaction is suitable in many applications, but the constraint of using rigid tools is not applicable to some situations, such as the use of catheters in virtual surgery, or of a rubber part in an assembly simulation. Rigid-tool-based interaction is also unable to provide force feedback regarding interaction through the human hand, due to the soft nature of human flesh. In this paper, we address some of the computational challenges of haptic interaction through deformable tools, which forms the basis for direct-hand haptic interaction. We describe a haptic rendering algorithm that enables interactive contact between deformable objects, including self-collisions and friction. This algorithm relies on a deformable tool model that combines rigid and deformable components, and we present the efficient simulation of such a model under robust implicit integration.

1. Introduction

Haptic rendering is the computational technology that allows us to interact with virtual worlds through the sense of touch. It relies on an algorithm that simulates a virtual world in a physically based manner and computes interaction forces, and on a robotic device that transmits those interaction forces to the user. Haptics science is a multidisciplinary field that brings together psychophysics research for the understanding of tactile cues and human perception; mechanical engineering for the design of robotic devices; control theory for the analysis of the coupling between the real and virtual worlds; and computer science, in particular computer graphics, for the simulation of the virtual world and the design of the haptic rendering algorithm [29].

In this paper, we focus on the haptic rendering of objects that are composed of both rigid and deformable parts. Many of the objects in the real world, including our own flesh, fit this description. In particular, our haptic rendering algorithm serves as the main building block for model-based computation of direct-hand haptic interaction. The hand is the main tactile sensor used by humans to capture information from the world [25]. It allows us to *manipulate* the world and provides us with two-way interaction with our environment. Despite the importance of the human hand for haptic interaction, current haptic devices and haptic rendering techniques suffer important limitations that have not allowed hand-based virtual touch to reach its full potential. Haptic rendering is typically carried out either through a pen-like robotic device, or through vibrotactile devices. In addition to hardware limitations, most haptic rendering algorithms are limited to point-based interaction between the user and the objects in the environment. There are some notable exceptions, both devices (e.g., exoskeleton structures [23,7,33]) and object-object rendering algorithms, also called 6-Degree-of-Freedom (6-DoF) haptic rendering [31,39,6].

We follow a strategy for haptic rendering of tool-based contact where a virtual replica of the tool is simulated in a virtual world, and haptic interaction takes place through a viscoelastic coupling between the haptic device and the tool [11]. Following this strategy, direct-hand touch can be rendered by simulating a virtual hand as the tool (as shown in Fig. 1), and coupling the device to this simulated hand. In Section 3, we summarize the adaptation of a multirate 6-DoF haptic rendering algorithm to model multipoint contact with objects composed of rigid and deformable parts, such as a hand. This algorithm does not suffer from the limitations of traditional single-point contact models for capturing the interaction between soft fingers and the environment.

In Section 4, taking the hand as an example model, we describe how to connect rigid and deformable components and how to actuate them through standard haptic devices. We exploit the position and orientation tracked by a haptic device to guide the rigid components of the virtual hand, and we model the coupling between the rigid and deformable components of the hand using stiff connections and implicit integration. As a first step toward full-hand haptic rendering, we consider the hand to be a shape formed by one rigid component and elastic flesh, and we do not take into account the articulations of the fingers.

A computational algorithm, described in Section 5, allows us to simulate a virtual hand using existing implementations of rigid-body and deformable-body simulations as black boxes. The use of

* Corresponding author.
E-mail addresses: carlos.garre@urjc.es (C. Garre), miguel.otaduy@urjc.es (M.A. Otaduy).

0097-8493/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2010.08.006

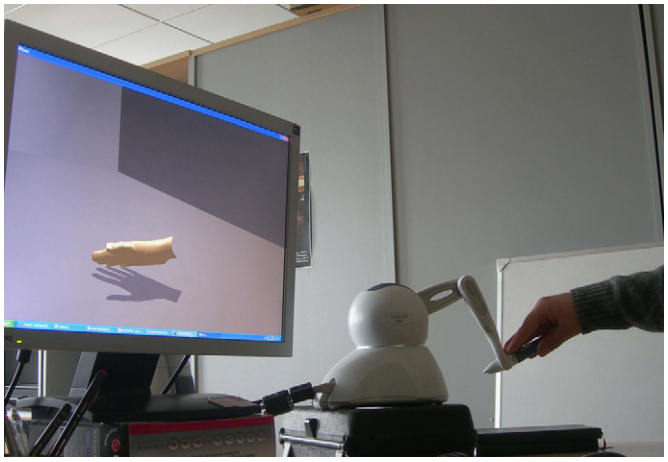

Fig. 1. Haptic manipulation of a virtual hand.

implicit integration couples the degrees of freedom of the rigid and deformable components of the virtual hand, but our algorithm allows us to work around these couplings and still use existing implementations as black boxes.

We show our haptic rendering algorithm applied to the interaction between a hand model and other complex deformable objects with friction (see Fig. 5), with several moving deformable objects (see Fig. 6), and with self-collisions between the fingers (see Fig. 3). In Section 6, we also analyze the impact of various parameters of the virtual scene on the computational cost of our haptic rendering algorithm.

2. Related work

The rendering of haptic interaction with interesting virtual environments relies to a large extent on the ability to simulate efficiently effects such as deformations and contact among the objects in the virtual environment. We refer the reader to surveys on those topics for more information [18,35,28,47]. The efficiency of the techniques for solving contact and deformation problems is growing steadily (see [5,45,20,22,43,40] for some recent examples), but the complexity of the scenes that can be handled at force-update rates is still far from desirable. The quality of the force feedback can be described as the ability to transparently convey contact and inertial forces during manipulation, while guaranteeing stable interaction. The range of contact impedances that can be rendered in a stable manner depends, however, on the force-update rate [10], and this sets a major challenge when simulating complex deformations. Measurement-based approaches [26] may decrease the high update-rate requirements, but it is currently still unclear how to extend such approaches to model sliding, rolling, or distributed contact between time-varying objects.

The fundamental approach for modeling complex contact is to maximize the efficiency of collision detection, dynamics simulation, and collision response algorithms. In the context of haptic rendering, researchers have proposed solutions for fast collision detection and response between rigid bodies [31,24,21], and even collision response between deformable objects [15,6] (although limited to certain types of deformations). Others have proposed level-of-detail algorithms that select the appropriate object resolution based on contact properties and the available computational time, for either rigid bodies [38] or deformable bodies [6].

Nevertheless, all such techniques may be limited by scene complexity and are unable to deliver high-quality feedback forces at desirable rates. Those techniques are complementary to our work, because we aim at combining a (possibly slow) visual simulation of the virtual environment with (fast) haptic rendering of the contact and deformation forces. Our work lies closer to earlier approaches that create at run-time a simplified model of contact and deformations, then use this model for feedback force updates.

Some multirate rendering approaches define active contact constraints in the slow thread and use them to compute contact forces in the fast thread through Signorini's contact model [14], penalty-based methods [39], projection of unconstrained accelerations [36], or a least-squares solution to Poisson's restitution hypothesis for rigid bodies [12]. Others perform approximate local updates of the contact constraints in the fast loop and then apply penalty-based forces [21]. Early approaches to multirate simulation of deformable models considered force extrapolation [41], meshes of different resolution coupled through Norton equivalents [2], or local linearized submeshes for approximating high-frequency behavior [8]. It is also worth noting existing work on stability analysis of multirate rendering algorithms [4].

The simplest way of coupling a slow-update simulation and fast-update force computation is the use of a virtual coupling [11] between the haptic device and a simulation of the handle. Stable rendering can be obtained by designing a passive simulation of the virtual environment and tuning the coupling parameters appropriately. Though stable, the transparency of the rendering may be highly compromised, because the maximum coupling stiffness may be rather low. In a multirate approach, however, the slow-computing simulation also generates an approximate intermediate representation of the interaction with the environment [1], and this intermediate representation is evaluated for force feedback. High rendering stiffness is possible in the multirate approach thanks to the intermediate representation, but rendering quality depends on the quality of the approximations.

The focus of our work is on the ability to interact haptically through objects composed of both rigid and deformable parts. The most prominent example is our own bone and flesh, and our haptic rendering algorithm serves as the basis for haptic interaction through the hand. At the current stage of our research, we use a rather approximate hand model, with soft tissue modeled using linear co-rotational finite elements [34] and coupled to rigid components. More sophisticated hand models exist in computer graphics. One possibility is to employ several captured poses of the hand and follow a pose-space deformation approach to synthesize arbitrary hand poses based on internal joint angles [27]. Another possibility is to employ biomechanical models that account for internal bone, tendon, and tissue interaction [46].

Classical approaches to haptic rendering of contact between the hand and objects in the virtual environment have modeled finger–object contact as a single contact point. The fingertip is modeled as a soft tissue, and contact forces are computed using Hertz contact theory [13]. This approach exploits point-based haptic rendering methods, also called 3-DoF haptic rendering [49,42].

In contrast to point-based rendering methods, in this work we advocate simulating a virtual model of the complete hand, computing contact between the virtual hand and the objects in the scene, and providing force feedback to the user exploiting the concept of virtual coupling [11].

Our work bears some similarity to the research of Otaduy and Gross [37] and of Duriez et al. [14,15]. Otaduy and Gross linearized the summed contact force and torque acting on a rigid tool w.r.t. other forces and torques (e.g., due to virtual coupling with the device). Instead, we build on the approach of Garre and Otaduy [16], which captures a (locally) linear representation of the contact between a deformable tool (e.g., the hand model) and other deformable objects, as well as the internal force between the deformable part of the tool and a rigid handle. This method is

qualitatively distinct, as it allows the user to manipulate a deformable tool. The approach followed to obtain the linear representation is similar to that of Otaduy and Gross, but ours accounts for much more complex contact between deformable objects. Duriez et al. proposed a method that identifies active constraints in the visual loop, and then solves a constrained problem with equality constraints in the haptic loop. Contact forces are transported to the rigid part of the tool, which is connected to the haptic device through virtual coupling. Their approach relies on quasi-rigid deformations; hence it cannot support large deformations. It also suffers an increase of the computational cost (in the haptic loop) as the number of contacts grows.

This paper is based on our preliminary work presented in [17]. Some of the key contributions we present in this paper are the computation of constrained dynamics for a tool composed of rigid and deformable parts, and the efficient computation of a linearized contact model for multirate rendering, subject to the coupling of rigid and deformable parts. Those elements were not covered in [17], and they are key for obtaining (i) highly realistic visual simulation of contact for tool objects made of rigid and deformable parts such as the human hand, and (ii) haptic feedback with a wide impedance range, even when the visual simulation runs at frequencies well under the haptic update rate. Another addition in this paper is an analysis of the computational cost of the different stages of our algorithm.

3. Overview of the haptic rendering algorithm

Haptic rendering typically distinguishes interaction through a tool object from direct interaction with the hand. Tool-based rendering algorithms simulate a virtual model of the tool and compute contact between the tool and other objects in the environment using robust contact-modeling algorithms. Direct-hand rendering algorithms, on the other hand, track points on the finger tips and model contact forces based on single-point penalty-based models (see [13] for an example). Here, we present a haptic rendering algorithm that allows the use of deformable tools, and we implement a human hand model as an example of such deformable tools. The key element in our algorithm is to simulate a virtual hand model in an analogous way to the simulation of a virtual tool, and to employ robust contact modeling methods to compute the interaction between the virtual hand and other virtual objects. The benefits of our algorithm include the possibility of modeling rich and complex frictional contact or even handle self-collisions between fingers (as shown in Fig. 3). Although here we show our model's application to the example of a human hand, our algorithm can be applied to various types of tools containing rigid and deformable parts.

3.1. Linking the device, the handle, and the tool

We assume that the haptic device operates in impedance mode, i.e., it tracks the configuration of the hand and controls the applied force. We define the configuration of the hand as a set of rigid frames, i.e., position and orientation. This definition is applicable, for example, to pen-like haptic devices, which track a single rigid frame that corresponds to the global configuration of the hand, or to glove-like devices that track the rigid configurations of several phalanxes.

We associate one rigid component in the virtual hand model with each rigid frame tracked, and we refer to this rigid component as a *handle*. In the case of a pen-like device, the handle is placed at the palm of the hand, as shown in Fig. 2-top; and in glove-like devices the handles can be placed at the phalanxes of the virtual hand model. For the rest of this section, we will describe a situation with a single handle, as in Fig. 2. As is also shown in Fig. 2-top, we set a one-directional viscoelastic virtual coupling between the tracked rigid frame that represents the configuration of the haptic device and the rigid handle in the virtual hand. The virtual coupling ensures that the handle follows the position and orientation tracked in the real world.

The majority of the hand is made of soft tissue; hence, we model the hand as a deformable object. Following the terminology in haptic rendering, the virtual hand model is denoted as *tool*, as it is the virtual object commanded by the haptic device. In our model, we connect the deformable hand or tool and the rigid handle with stiff springs. In Fig. 2-top, $\mathbf{F}_c$ denotes the total coupling force between handle and tool. Note that the inertial effects of the hand or of the contact between the hand and other objects in the virtual environment are transmitted to the handle through the coupling force $\mathbf{F}_c$.

3.2. Constraint-based simulation

As part of the haptic rendering algorithm, we solve a constrained dynamics problem involving the deformable hand, the rigid handle, and possibly other dynamic objects in the virtual environment. The constrained dynamics solver requires an efficient solution of the coupled dynamics of the tool-hand and the handle. The details of the constrained dynamics solver have been published elsewhere [40], but here we outline its application to the solution of the coupled dynamics of the complete hand model.

We denote with subindices *h*, *t*, and *e* quantities related to the *handle*, *tool*, and *environment*, respectively. The equations of motion for the handle, the tool and the environment can be

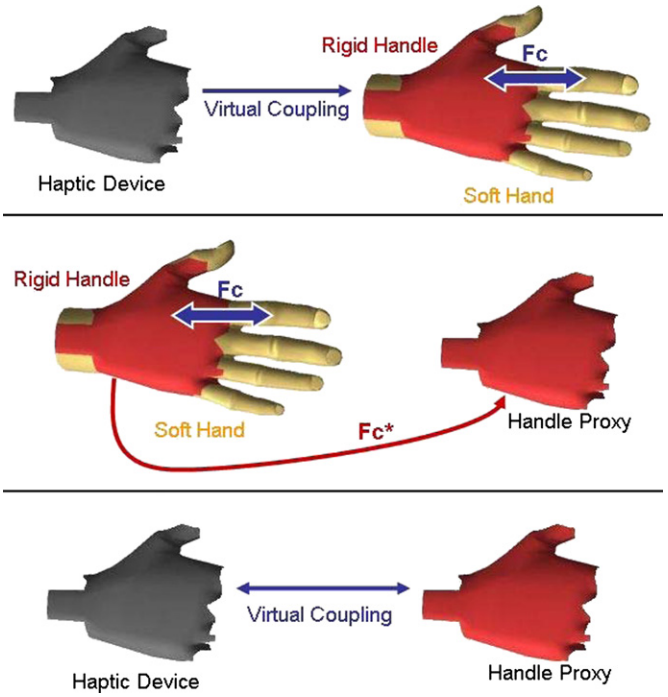

Fig. 2. Basic steps of haptic rendering based on handle-space force linearization [16]. Top: the haptic device is connected to a rigid handle on the hand model, which is in turn coupled through stiff springs to the rest of the soft hand. Middle: a proxy of the handle receives a linearized approximation $\mathbf{F}_c^*$ of the actual coupling forces $\mathbf{F}_c$ between the handle and the rest of the hand. Bottom: the feedback force is computed through a two-way virtual coupling between the haptic device and the proxy handle.

written as

$$\mathbf{M}_h\dot{\mathbf{v}}_h = \mathbf{F}_h,$$

$$\mathbf{M}_t\dot{\mathbf{v}}_t = \mathbf{F}_t + \mathbf{J}_t^T\lambda,$$

$$\mathbf{M}_e\dot{\mathbf{v}}_e = \mathbf{F}_e + \mathbf{J}_e^T\lambda, \tag{1}$$

and the contact constraints can be written as

$$\mathbf{J}_t\mathbf{v}_t + \mathbf{J}_e\mathbf{v}_e \geq \mathbf{b}_\lambda. \tag{2}$$

Here, $\mathbf{J}_t^T\lambda$ represent constraint forces acting on the tool, and $\mathbf{J}_e^T\lambda$ represent constraint forces acting on the environment. Contact constraints do not act directly on the handle, as we assume that it is an internal, rigid part of the hand.

We discretize the dynamics equations using implicit integration [3], which allows the use of large time steps even with stiff forces. In our case, we have used the semi-implicit Euler method with linearized forces. Implicit integration yields a linear system of equations of the form:

$$\mathbf{A}_h\mathbf{v}_h + \mathbf{A}_{ht}\mathbf{v}_t = \mathbf{b}_h,$$

$$\mathbf{A}_{th}\mathbf{v}_h + \mathbf{A}_t\mathbf{v}_t = \mathbf{b}_t + \mathbf{J}_t^T\lambda,$$

$$\mathbf{A}_e\mathbf{v}_e = \mathbf{b}_e + \mathbf{J}_e^T\lambda,$$

$$\mathbf{J}_t\mathbf{v}_t + \mathbf{J}_e\mathbf{v}_e \geq \mathbf{b}. \tag{3}$$

In this system, the matrices $\mathbf{A}_h$ and $\mathbf{A}_t$ are related to the uncoupled dynamics of the hand and the handle. The matrix $\mathbf{A}_h$ is a dense 6×6 matrix, while $\mathbf{A}_t$ is a sparse, symmetric, positive definite, $3n\times 3n$ matrix, with n the number of nodes in the tool-hand. The terms $\mathbf{A}_{ht}$ and $\mathbf{A}_{th}$ arise due to the implicit integration of the coupling between hand and handle. The matrix $\mathbf{A}_e$ represents the dynamics of the environment and is of arbitrary size.

The problem above can be regarded as a *mixed linear complementarity problem*, which, as discussed in [40], is equivalent to solving a *linear complementarity problem* of the type:

$$\mathbf{A}_\lambda\lambda \geq \mathbf{b}_\lambda, \quad \lambda \geq 0. \tag{4}$$

For the exact formulation of the matrix $\mathbf{A}_\lambda$ and the vector $\mathbf{b}_\lambda$, refer to [40].

3.3. Linearized tool-handle coupling for multirate rendering

If we want realistic and high-impedance feedback, we need a high update rate of feedback forces (e.g., 1 kHz). At such a high rate, it is not possible to compute all the tasks involved in haptic rendering: collision detection and response, simulation of the virtual environment, and elastic deformations. Hence, we use a multirate rendering algorithm based on linearization of the coupling force in handle space [16]. This algorithm simulates in a visual loop the constrained dynamics of the hand model as described above, and computes in a haptic loop the forces to be transmitted to the user.

In addition to computing the motion of the hand, in the visual loop, we also compute a linearized version of the coupling forces between the handle and the rest of the tool-hand. In Fig. 2-center, $\mathbf{F}_c^*$ denotes the linearized coupling force. The details of the linearization are described in [16], but to summarize, it accounts for the change in $\mathbf{F}_c$ as a function of the handle and tool velocities, which are in turn expressed as a function of the handle's state (i.e., position and velocity) at the beginning of a simulation step. In other words, the linearized force is expressed as

$$\mathbf{F}_c^*(\mathbf{x}_h,\mathbf{v}_h) = \mathbf{F}_c(\mathbf{x}_0,\mathbf{v}_0) + \frac{d\mathbf{F}_c}{d\mathbf{x}_{h,0}}(\mathbf{x}_h-\mathbf{x}_0) + \frac{d\mathbf{F}_c}{d\mathbf{v}_{h,0}}(\mathbf{v}_h-\mathbf{v}_0), \tag{5}$$

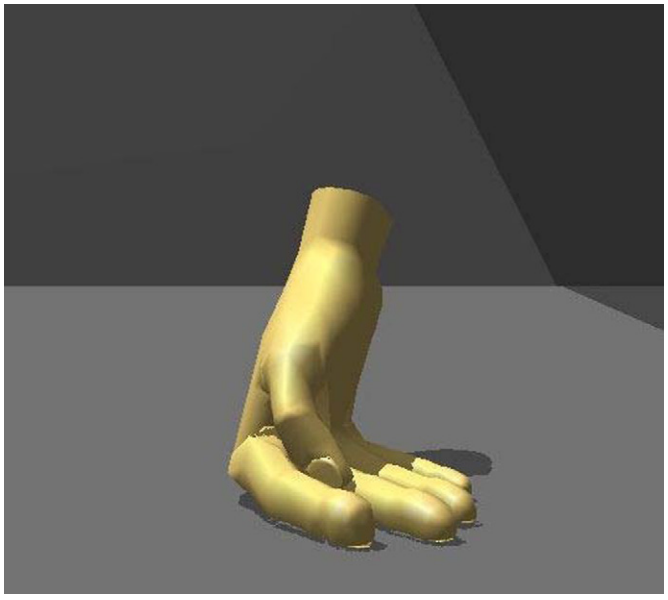

Fig. 3. Self-collision between thumb and other fingers.

and the derivative terms are computed as

$$\frac{d\mathbf{F}_c}{d\mathbf{x}_{h,0}} = \frac{\partial\mathbf{F}_c}{\partial\mathbf{x}_{h,0}} + \frac{\partial\mathbf{F}_c}{\partial\mathbf{v}_h}\frac{\partial\mathbf{v}_h}{\partial\mathbf{x}_{h,0}} + \frac{\partial\mathbf{F}_c}{\partial\mathbf{v}_t}\frac{\partial\mathbf{v}_t}{\partial\mathbf{x}_{h,0}},$$

$$\frac{d\mathbf{F}_c}{d\mathbf{v}_{h,0}} = \frac{\partial\mathbf{F}_c}{\partial\mathbf{v}_h}\frac{\partial\mathbf{v}_h}{\partial\mathbf{v}_{h,0}} + \frac{\partial\mathbf{F}_c}{\partial\mathbf{v}_t}\frac{\partial\mathbf{v}_t}{\partial\mathbf{v}_{h,0}}. \tag{6}$$

The key difficulty in the computation of the linearized coupling force lies in the terms $(\partial\mathbf{v}_h/\partial\mathbf{x}_{h,0}, \partial\mathbf{v}_h/\partial\mathbf{v}_{h,0}, \partial\mathbf{v}_t/\partial\mathbf{x}_{h,0}, \partial\mathbf{v}_t/\partial\mathbf{v}_{h,0})$, which depend on the actual active constraints in the problem from Eq. (4) and are affected by the coupling between rigid handle and deformable tool. The details of the computation of these terms are given later in Section 5.3.

While the visual loop computes the simulation of the full scene, in the haptic loop we simulate a proxy of the rigid handle. This proxy handle is connected through a two-way viscoelastic virtual coupling to the input rigid frame of the haptic device. However, instead of simulating the complete interaction between the handle and the rest of the virtual hand, we approximate it using the linearized version of the coupling. This linearized coupling encapsulates effects such as contact between the hand and other objects in the environment; therefore, there is no need for expensive collision detection or simulation of deformations and contact in the haptic loop. Finally, the feedback force sent to the user is computed through the virtual coupling between the input haptic device frame and the proxy handle, as shown in Fig. 2-bottom. Appropriate force feedback requires that the haptic device contain a force actuator associated with each tracked rigid frame. In other words, a pen-like device should provide one global force, and a glove-like device should provide a force for each tracked phalanx.

4. Hand model with rigid and deformable parts

To model the elasticity of the hand (our deformable tool), we use a linear co-rotational finite element model [34]. To apply the haptic rendering algorithm described in the previous section, we need to identify a rigid handle for each rigid frame tracked by the haptic device. In our case, using the Phantom Omni as the haptic device, we need one rigid handle. We choose to locate the rigid handle at the palm of the hand model. We need to define the

rigid-body properties of the handle, but we do not want to sculpt a separate 3D model. Therefore, given the tetrahedral mesh of the whole hand, we define the rigid handle by selecting a seed point in the palm and collecting all hand nodes inside a sphere centered at the seed. Fig. 4 depicts a 2D example of hand and handle design.

4.1. Dynamics equations with rigid and deformable parts

The state $\mathbf{x}_t \in \mathbb{R}^{3n}$ and velocity $\mathbf{v}_t \in \mathbb{R}^{3n}$ of the deformable hand (a.k.a. the tool) are defined by the positions and velocities of all the n nodes that form the tetrahedral mesh used for the dynamic simulation. The state $\mathbf{x}_h \in \mathbb{R}^7$ of the handle, on the other hand, comprises the position of its center of mass $\mathbf{x}_{com}$ and its orientation, defined as a quaternion $\mathbf{q}_h$. The velocity $\mathbf{v}_h \in \mathbb{R}^6$ of the handle comprises the velocity of the center of mass $\mathbf{v}_{com}$ and the angular velocity $\boldsymbol{\omega}_h$. For clarity, here we summarize all state and velocity vectors as column vectors:

$$\mathbf{x}_t = (\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_n^T)^T, \quad \mathbf{x}_h = (\mathbf{x}_{com}^T, \mathbf{q}_h^T)^T,$$

$$\mathbf{v}_t = (\mathbf{v}_1^T, \mathbf{v}_2^T, \ldots, \mathbf{v}_n^T)^T, \quad \mathbf{v}_h = (\mathbf{v}_{com}^T, \boldsymbol{\omega}_h^T)^T. \tag{7}$$

The mass matrix of the tool hand $\mathbf{M}_t$ is defined by lumping masses at the nodes, i.e.,

$$\mathbf{M}_t = \begin{pmatrix} m_1\mathbf{I}_{3\times3} & 0 & \ldots & 0 \\ 0 & m_2\mathbf{I}_{3\times3} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \ldots & 0 & m_n\mathbf{I}_{3\times3} \end{pmatrix}. \tag{8}$$

The mass matrix of the handle is defined as

$$\mathbf{M}_h = \begin{pmatrix} m_h\mathbf{I}_{3\times3} & 0 \\ 0 & \mathbf{M}_q \end{pmatrix}, \tag{9}$$

where m_h is the total mass of the rigid handle, and $\mathbf{M}_q$ is its inertia matrix. We compute the mass and inertia properties of the handle from the lumped masses of the selected hand nodes [32].

To couple the hand and the handle, we set stiff damped springs between pairs of corresponding nodes of both models. The springs model two-way coupling as a vector of internal forces F_c. Given that the hand and handle are co-located when they are created, these springs have zero rest length. This fact largely simplifies the formulation of the spring force, which becomes linear in the positions of the end points. Let us consider a spring between a node $\mathbf{x}_A$ of the hand and a point $\mathbf{x}_B$ on the handle (as shown in Fig. 4), with velocities $\mathbf{v}_A$ and $\mathbf{v}_B$. Then, the force $\mathbf{F}_A$ acting on the node of the hand can be written as

$$\begin{aligned} &\mathbf{F}_A = -k(\mathbf{x}_A - \mathbf{x}_B) - d(\mathbf{v}_A - \mathbf{v}_B) \\ &\text{with } \mathbf{x}_B = \mathbf{x}_{com} + \mathbf{R}_h\mathbf{r}_B \\ &\text{and } \mathbf{v}_B = \mathbf{v}_{com} + \boldsymbol{\omega}_h \times (\mathbf{R}_h\mathbf{r}_B). \end{aligned} \tag{10}$$

Note that the position and velocity of the point $\mathbf{x}_B$ are expressed in terms of the handle state and velocity vectors, with $\mathbf{R}_h$ a rotation matrix that represents the orientation of the handle, and $\mathbf{r}_B$ the position of $\mathbf{x}_B$ in the local reference frame of the handle. The coefficients k and d are the stiffness and damping parameters of the spring.

Likewise, the opposite force acting on the handle is $\mathbf{F}_B = -\mathbf{F}_A$, while the torque can be written as

$$\mathbf{T}_B = (\mathbf{R}_h\mathbf{r}_B) \times \mathbf{F}_B. \tag{11}$$

In order to ensure stiff yet stable coupling between the hand and the handle, we integrate the dynamics equations using implicit integration methods. Without coupling springs, the dynamics equations of the hand and the handle can be solved separately, but the coupling also couples the systems of equations. In Section 5, we present an algorithm for solving the coupled hand and handle dynamics.

4.2. Coupling terms under implicit integration

Implicit integration of the dynamics yields Eq. (3). We will focus here on the terms $\mathbf{A}_{ht}$ and $\mathbf{A}_{th}$, which arise due to the coupling between hand and handle.

With implicit Euler, the matrix $\mathbf{A}_{ht}$ is formulated as

$$\mathbf{A}_{ht} = \mathbf{M}_{ht} - \Delta t \frac{\partial \mathbf{F}_h}{\partial \mathbf{v}_t} - \Delta t^2 \frac{\partial \mathbf{F}_h}{\partial \mathbf{x}_t}. \tag{12}$$

Here, the term $\mathbf{M}_{ht} = 0$ because there is no mass coupling between handle and hand. The term $\partial\mathbf{F}_h/\partial\mathbf{v}_t$ represents the Jacobian of the coupling forces (and torques) acting on the handle w.r.t. hand velocities. We leave this term to the reader, as it is less complex than the term $\partial\mathbf{F}_h/\partial\mathbf{x}_t$, which represents the Jacobian of the coupling forces acting on the handle w.r.t. the hand state. This last term can be decomposed in a block representation as follows:

$$\frac{\partial \mathbf{F}_h}{\partial \mathbf{x}_t} = \begin{pmatrix} \frac{\partial \mathbf{F}_h}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{F}_h}{\partial \mathbf{x}_n} \end{pmatrix}. \tag{13}$$

Each 6×3 submatrix captures the Jacobian of the coupling force and torque acting on the handle w.r.t. the position of one node of the tool hand. Then, the terms for hand nodes $\{\mathbf{x}_i\}$ that are not connected by springs to the handle are $\partial\mathbf{F}_h/\partial\mathbf{x}_i = 0$, and the non-zero terms capture the Jacobian of one and only one spring.

The matrix $\mathbf{A}_{th}$ in Eq. (3) is computed similarly as

$$\mathbf{A}_{th} = \mathbf{M}_{th} - \Delta t \frac{\partial \mathbf{F}_t}{\partial \mathbf{v}_h} - \Delta t^2 \mathbf{G} \frac{\partial \mathbf{F}_t}{\partial \mathbf{x}_h}. \tag{14}$$

Here, the matrix $\mathbf{G}$ relates the angular velocity of the handle to the derivative of its orientation (represented as a quaternion in our case) [44]. The term $\partial\mathbf{F}_t/\partial\mathbf{x}_h$ is formed by rows $\partial\mathbf{F}_i/\partial\mathbf{x}_h$, which represent the Jacobian of the coupling force acting on nodes $\mathbf{x}_i$ of the hand w.r.t. the velocity vector of the handle. As in the observation discussed earlier, a non-zero term $\partial\mathbf{F}_i/\partial\mathbf{x}_h$ refers to one and only one coupling spring between the hand and the handle.

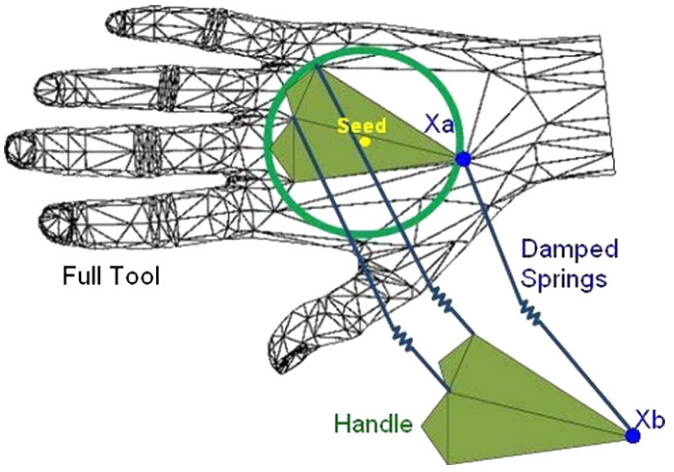

Fig. 4. Given the mesh of the deformable hand, we construct the rigid handle (in green) by picking a seed point in the palm and collecting all hand nodes inside a sphere centered at the seed. We couple the hand and the handle using stiff damped springs (in blue). Note that the hand and the handle are co-located when they are created, hence the coupling springs have zero rest length. In the figure, the hand and handle are artificially separated for clarity. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

From the above derivation, the assembly of the matrices $\mathbf{A}_{ht}$ and $\mathbf{A}_{th}$ is reduced to computing the Jacobians of the coupling spring forces described in Section 4. Recall that $\mathbf{F}_A$ in Eq. (10) represents a spring force acting on the node $\mathbf{x}_A$ of the tool hand, $\mathbf{F}_B=-\mathbf{F}_A$ represents the force acting on the handle, and $\mathbf{T}_B$ in Eq. (11) represents the spring torque acting on the handle. The Jacobians in $\partial\mathbf{F}_h/\partial\mathbf{x}_A$ and $\partial\mathbf{F}_A/\partial\mathbf{x}_h$ can then be computed as

$$\frac{\partial\mathbf{F}_B}{\partial\mathbf{x}_A}=k\mathbf{I},$$

$$\frac{\partial\mathbf{T}_B}{\partial\mathbf{x}_A}=(\mathbf{R}_h\mathbf{r}_B)\times\frac{\partial\mathbf{F}_B}{\partial\mathbf{x}_A},$$

$$\frac{\partial\mathbf{F}_A}{\partial\mathbf{x}_{com}}=k\mathbf{I},$$

$$\frac{\partial\mathbf{F}_A}{\partial\mathbf{q}_h}=k\frac{\partial(\mathbf{R}_h\mathbf{r}_B)}{\partial\mathbf{q}_h}+d\cdot\boldsymbol{\omega}_h\times\frac{\partial(\mathbf{R}_h\mathbf{r}_B)}{\partial\mathbf{q}_h}. \tag{15}$$

With a handle quaternion defined as $\mathbf{q}_h=(\mathbf{u}_h,s_h)$, the derivative of the rotated vector $\mathbf{R}_h\mathbf{r}_B$ w.r.t. the quaternion can be computed as

$$\frac{\partial(\mathbf{R}_h\mathbf{r}_B)}{\partial\mathbf{q}_h}=(\mathbf{r}_B^T\mathbf{u}_h\mathbf{I}+\mathbf{u}_h\mathbf{r}_B^T-2s_h\mathbf{r}_B^*+(\mathbf{r}_B\times\mathbf{u}_h)^*-\mathbf{u}_h^*\mathbf{r}_B^*,\ \ 2s_h\mathbf{r}_B+2\mathbf{u}_h\times\mathbf{r}_B), \tag{16}$$

where $\mathbf{r}_B^*$ is the skew-symmetric matrix that represents a cross product $\mathbf{r}_B\times$ as a linear transformation.

5. Implicit solution for rigid and deformable parts

As part of the haptic rendering algorithm, we need to solve a constrained dynamics problem involving the deformable hand, the rigid handle, and possibly other dynamic objects in the virtual environment. Then, the coupling force between the tool-hand and the handle must be linearized w.r.t. the handle state, accounting for contact constraints and inertial effects. Both the constrained dynamics solver and the linearization require an efficient solution to the coupled dynamics of the tool-hand and the handle. Here, we describe in detail our novel algorithm for efficiently solving the coupled dynamics of the complete hand model in the three following settings: unconstrained motion, constrained motion, and coupling force linearization.

5.1. Resolution of the unconstrained system

At every time step of the dynamics simulation, the computation of the unconstrained handle and tool-hand velocities requires the solution to the linear system in Eq. (3) (without the contact constraints). One possibility could be to formulate one large system containing all equations and the use of a state-of-the-art linear system solver to compute all velocities at once. However, this possibility suffers from two disadvantages: (i) the matrices $\mathbf{A}_{ht}$ and $\mathbf{A}_{th}$ might be rather dense if the handle is large (i.e., it contains many nodes of the hand), thereby complicating the use of solvers for sparse systems, and (ii) the choice of solver would require knowing the particular characteristics of the matrices $\mathbf{A}_h$ and $\mathbf{A}_t$, and hence the implementations of rigid body simulation (for the handle) and deformable body simulation (for the hand) could not be interpreted as black boxes in the coupled implementation, thereby complicating the job of the programmer.

Instead, we propose a solution to Eq. (3) that can treat the rigid-body simulation and the deformable-body simulation as black boxes, and does not require implementing additional linear-system solvers. Our solution is based on a sequential solution of $\mathbf{v}_t$ and $\mathbf{v}_h$ based on the computation of a Schur complement [19] in Eq. (3). We first single out $\mathbf{v}_t$ in the second row of Eq. (3) as

$$\mathbf{v}_t=\mathbf{A}_t^{-1}(\mathbf{b}_t-\mathbf{A}_{th}\mathbf{v}_h). \tag{17}$$

We then substitute the result in the first row, which yields the following linear system:

$$(\mathbf{A}_h-\mathbf{A}_{ht}\mathbf{A}_t^{-1}\mathbf{A}_{th})\mathbf{v}_h=\mathbf{b}_h-\mathbf{A}_{ht}\mathbf{A}_t^{-1}\mathbf{b}_t. \tag{18}$$

Here, $\mathbf{A}_h-\mathbf{A}_{ht}\mathbf{A}_t^{-1}\mathbf{A}_{th}$ is the Schur complement of $\mathbf{A}_t$. The velocity of the handle, $\mathbf{v}_h$, can be solved through Gaussian elimination in Eq. (18). Then, the solution is plugged into Eq. (17), and we solve for the velocity of the tool, $\mathbf{v}_t$. Note that we do not invert $\mathbf{A}_t$ in Eq. (17); we use a sparse linear system solver.

The reader may observe that there are two options for solving Eq. (3) in a sequential manner using Schur complements. Our choice, which formulates the Schur complement of $\mathbf{A}_t$, requires the solution to 8 large linear systems of the form $\mathbf{A}_t\mathbf{s}=\mathbf{r}$: 1 in the computation of the right-hand side in Eq. (18), 6 in the computation of $\mathbf{A}_t^{-1}\mathbf{A}_{th}$ (one per column of $\mathbf{A}_{th}$), and 1 in the final computation of $\mathbf{v}_t$ in Eq. (17). We speed up these solutions by pre-computing a Cholesky factorization of $\mathbf{A}_t$ once every visual simulation step (with the TAUCS library [48]). The other choice, which formulates the Schur complement of $\mathbf{A}_h$, may suffer from dense matrices. The term $\mathbf{A}_{th}\mathbf{A}_h^{-1}\mathbf{A}_{ht}$ would contain a non-zero block for each pair of nodes of the hand connected to the handle, and its density would grow with the size of the handle. Our solution, on the other hand, scales linearly (as desired) with the number of nodes in the handle.

5.2. Resolution of the constrained system

The solution to the constrained problem follows a similar approach of Schur complement computation, but requires further steps. We first single out $\mathbf{v}_t$ in the second row of Eq. (3) as

$$\mathbf{v}_t=\mathbf{A}_t^{-1}(\mathbf{b}_t+\mathbf{J}_t^T\lambda-\mathbf{A}_{th}\mathbf{v}_h). \tag{19}$$

We then substitute the result in the first row, which yields the following linear system:

$$\tilde{\mathbf{A}}_h\mathbf{v}_h=\tilde{\mathbf{b}}_h+\tilde{\mathbf{J}}_h^T\lambda \quad \text{with}$$

$$\tilde{\mathbf{A}}_h=\mathbf{A}_h-\mathbf{A}_{ht}\mathbf{A}_t^{-1}\mathbf{A}_{th},$$

$$\tilde{\mathbf{b}}_h=\mathbf{b}_h-\mathbf{A}_{ht}\mathbf{A}_t^{-1}\mathbf{b}_t,$$

$$\tilde{\mathbf{J}}_h^T=\mathbf{A}_{ht}\mathbf{A}_t^{-1}\mathbf{J}_t^T. \tag{20}$$

The most efficient way to formulate the terms above is to first compute $\mathbf{A}_{ht}\mathbf{A}_t^{-1}$, which requires the solution to 6 large sparse linear systems.

Substituting $\mathbf{v}_h$ in Eq. (19), we obtain

$$\mathbf{A}_t\mathbf{v}_t=\tilde{\mathbf{b}}_t+\tilde{\mathbf{J}}_t^T\lambda \quad \text{with}$$

$$\tilde{\mathbf{b}}_t=\mathbf{b}_t-\mathbf{A}_{th}\tilde{\mathbf{A}}_h^{-1}\tilde{\mathbf{b}}_h,$$

$$\tilde{\mathbf{J}}_t^T=\mathbf{J}_t^T-\mathbf{A}_{th}\tilde{\mathbf{A}}_h^{-1}\tilde{\mathbf{J}}_h^T. \tag{21}$$

The formulation of this system of equations is fairly inexpensive, as the computation of $\tilde{\mathbf{A}}_h^{-1}$ implies only the inversion of a small 6×6 matrix. The possible drawbacks of the Schur complement approach arise in the modified constraint matrix $\tilde{\mathbf{J}}_t^T$. The more nodes of the tool are linked to the handle, the denser the modified constraint matrix.

As a result of the Schur complement computation, the constrained system in Eq. (3) can now be written as

$$\mathbf{A}_t\mathbf{v}_t=\tilde{\mathbf{b}}_t+\tilde{\mathbf{J}}_t^T\lambda,$$

$$\mathbf{A}_e\mathbf{v}_e = \mathbf{b}_e + \mathbf{J}_e^T\lambda,$$

$$\mathbf{J}_t\mathbf{v}_t + \mathbf{J}_e\mathbf{v}_e \geq \mathbf{b}_\lambda, \tag{22}$$

where the velocity of the handle, $\mathbf{v}_h$, has been eliminated from the equations. This new system is a standard mixed linear complementarity problem, which we solve following the method described in [40]. Once the contact forces λ are computed, we can prune the active constraints, and Eq. (4) turns into an equality-constrained problem

$$\mathbf{A}_\lambda\lambda = \mathbf{b}_\lambda. \tag{23}$$

Given λ, we solve for the tool and handle velocities by substitution into Eqs. (21) and (20):

$$\mathbf{v}_t = \mathbf{A}_t^{-1}(\tilde{\mathbf{b}}_t + \tilde{\mathbf{J}}_t^T\mathbf{A}_\lambda^{-1}\mathbf{b}_\lambda),$$

$$\mathbf{v}_h = \tilde{\mathbf{A}}_h^{-1}(\tilde{\mathbf{b}}_h + \tilde{\mathbf{J}}_h^T\mathbf{A}_\lambda^{-1}\mathbf{b}_\lambda). \tag{24}$$

5.3. Linearization of the coupling force

The computation of the linearized coupling force, outlined in Section 3.3, requires the derivative terms in Eq. (6). From this set of derivatives, $\partial\mathbf{F}_c/\partial\mathbf{x}_{h,0}$ is expressed in Eq. (15), and the derivatices w.r.t. velocities $\partial\mathbf{F}_c/\partial\mathbf{v}_h$ and $\partial\mathbf{F}_c/\partial\mathbf{v}_t$ can be obtained in a similar way. The rest of the terms ($\partial\mathbf{v}_h/\partial\mathbf{x}_{h,0}$,$\partial\mathbf{v}_h/\partial\mathbf{v}_{h,0}$,$\partial\mathbf{v}_t/\partial\mathbf{x}_{h,0}$,$\partial\mathbf{v}_t/\partial\mathbf{v}_{h,0}$) can be obtained by differentiation of Eq. (24), such as

$$\frac{\partial\mathbf{v}_h}{\partial\mathbf{x}_{h,0}} = \tilde{\mathbf{A}}_h^{-1}\left(\frac{\partial\tilde{\mathbf{b}}_h}{\partial\mathbf{x}_{h,0}} + \tilde{\mathbf{J}}_h^T\mathbf{A}_\lambda^{-1}\frac{\partial\mathbf{b}_\lambda}{\partial\mathbf{x}_{h,0}}\right). \tag{25}$$

These derivatives do not suffer particular complications induced by the handle-tool coupling, because the Schur complement formulation already deals with the coupling terms. In our implementation, we have approximated the derivatives by approximating the matrix $\mathbf{A}_\lambda$. This matrix is typically dense and large, but we consider only its diagonal terms, which amounts to considering different contacts to be inertially decoupled. Note that we only diagonalize $\mathbf{A}_\lambda$ in the computation of the linearized coupling model, *not* in the actual solution of constrained dynamics in the visual loop.

Computing the derivatives $\partial\mathbf{b}_\lambda/\partial\mathbf{x}_{h,0}$ requires solving several linear systems that involve $\mathbf{A}_t$. Specifically, we need to solve $2\times(6+7)=26$ linear systems in total. We speed up these solutions by reusing the Cholesky factorization of $\mathbf{A}_t$, as discussed in Section 5.1.

6. Experiments

We executed our experiments on a quad-core 2.4 GHz PC with 3 GB of memory (although we used only two processors, for the visual and haptic loops) and a GeForce 8800 GTS. We manipulated the models using a Phantom Omni haptic device from SensAble Technologies.

We tested our algorithm with some 3D scenarios consisting of one deformable tool (a model of a human hand, in most experiments) and a set of rigid and deformable objects. A video showing some of our experiments can be downloaded from http://www.gmrv.es/~cgarre/DivX_GarreOtaduyCG.avi.

Fig. 5 shows situations with large-area contact, intricate contact, and large deformation when touching a bunny model. For the bunny, we used a surface mesh with 4000 triangles for collision detection and a tetrahedral mesh with 271 tetrahedra for the deformation. Fig. 6 depicts several snapshots of a scene where the virtual hand interacts with moving letters. The letters were modeled with meshes with 175 triangles and 65 tetrahedra on average. Fig. 3, on the other hand, shows that we can also handle self-contact among fingers of the hand itself. We have also tested other deformable tools, such as an alien toy, another bunny, and a sword, as seen in Fig. 7.

To evaluate the computational cost of our algorithm, we obtained statistics from a simulation where the hand tool touches a single rigid object. The hand followed a prescribed trajectory to guarantee the same motion in all simulations. We collected results for three different sizes of the triangle mesh of the hand model (700, 2100, and 6300 triangles) and three different sizes of

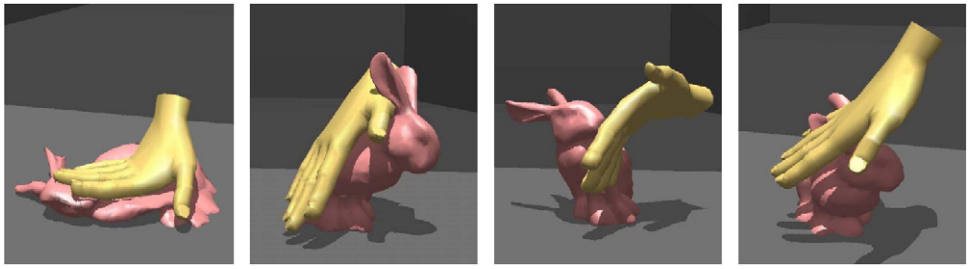

Fig. 5. Virtual hand touching a bunny model. Notice the large deformations (in particular in the leftmost figure), large-area contact, intricate contact (see the ear sticking out between the fingers in the rightmost figure), and friction effects, all computed interactively using our algorithm.

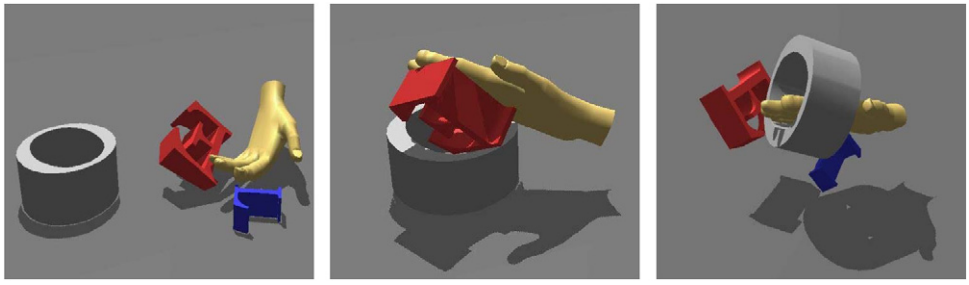

Fig. 6. Our haptic rendering algorithm also serves to touch moving objects (such as the letters) with a virtual hand.

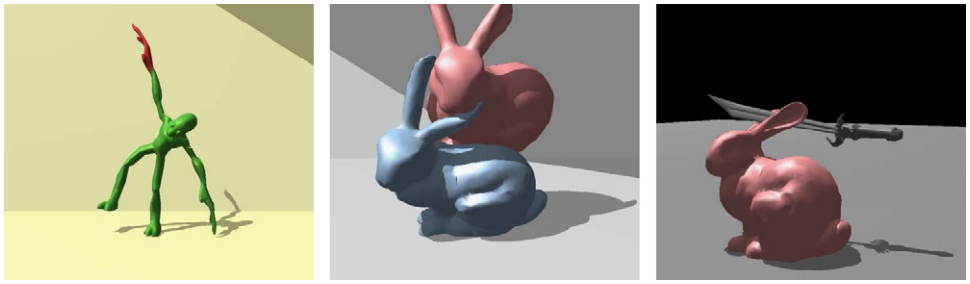

Fig. 7. We have tested various tools with rigid and deformable parts.

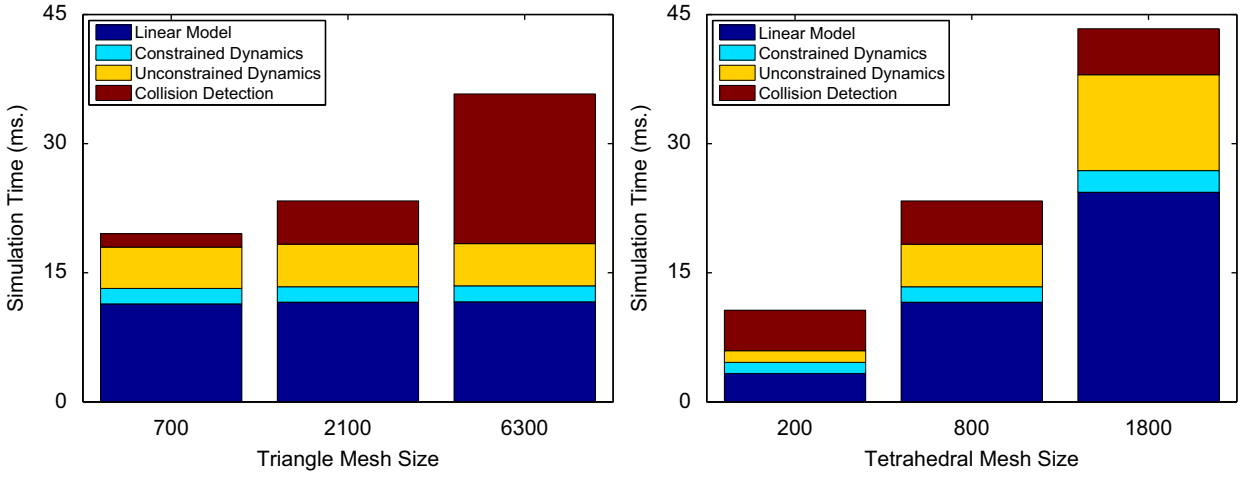

Fig. 8. Computational cost of the different stages of the algorithm. Left: varying the size of the triangular mesh. Right: varying the size of the tetrahedral mesh.

the tetrahedral finite element mesh (200, 800, and 1800 tetrahedra). We measured the average time spent in the different stages of the visual loop of our haptic rendering algorithm, i.e., collision detection, unconstrained dynamics, constrained dynamics, and linearized force model computation. With the finest discretizations, the simulation cost reaches levels that border the minimum admissible rate for the update of the linearized force model. Note that, despite the time spent in the visual loop, the haptic loop is guaranteed to run at 1 kHz in our multirate rendering algorithm.

Fig. 8 shows the average cost for the different stages of our solver. In Fig. 8-left, we compare the cost when using different triangle mesh sizes. It is clear that, thanks to a decoupling of the tetrahedral finite element mesh from the surface triangle mesh, the size of the triangle mesh affects only the cost of collision detection, and the cost of the rest of the algorithm remains fixed. In Fig. 8-right, we compare the cost of using different tetrahedral mesh sizes. In this case, the situation is almost the opposite. The number of tetrahedra does not affect the collision detection stage, but it clearly affects the rest of the algorithm, in particular the computation of the linearized model and the unconstrained dynamics solution. The cost of these two stages grows approximately linearly with the number of tetrahedra. In all simulations, the rigid handle covered 100 nodes of the tetrahedral tool mesh.

The visual simulation runs at an average of 50 fps in situations with contact, while, as mentioned earlier, feedback forces are computed at a frequency of 1 kHz. To a large extent, the high efficiency of the constrained dynamics simulation is obtained due to our efficient handling of the implicit coupling between the rigid and deformable components of the hand model.

7. Discussion and future work

We have presented a haptic rendering algorithm for computing force interaction through deformable tools such as a human hand. Our algorithm differs from previous rendering algorithms in that it employs a deformable tool model that couples rigid and deformable components, and we have presented a computational algorithm that efficiently solves implicit integration under such coupling, thus allowing effective haptic rendering.

Our proposed algorithm constitutes a step forward toward full-hand haptic interaction, but there are still unsolved problems. We plan to extend our implementation to hand models that combine soft tissue and an articulated model for the skeleton. The associated challenges deal mostly with the extension of virtual coupling approaches to the articulated skeleton. We will consider solutions for modeling skeletal posture statistically from a few tracked phalanxes [9] and control approaches for synthesizing posture from sparse data [30].

We are also interested in the evaluation of our model in terms of perception, in aspects such as sensory substitution from full-hand touch to a stylus-type device. In full object-object interaction, the tool is often constrained in orientation. Even though our model is able to compute torque feedback, our current 3-DoF device is not capable of transmitting this torque, and the user does not perceive orientation constraints. An even more serious problem due to the 3-DoF device is the possible lack of passivity, which may turn the simulation momentarily unstable under fast rotational contact.

We are also adapting our haptic rendering algorithm to exoskeleton haptic devices, which would enable tracking of and

force-feedback application to each finger independently. While designing our algorithm, we have encountered several alternatives for the specifics of the linearization, and we are currently analyzing these alternatives.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments, and the GMRV group at URJC. This work was funded in part by the URJC - Comunidad de Madrid Project CCG08-URJC/DPI-3647 and by the Spanish Science and Innovation Dept. Projects TIN2009-07942 and PSE-300000-5.

Appendix A. Supplementary material

Supplementary data associated with this article can be found in the online version of 10.1016/j.cag.2010.08.006.

References

[1] Adachi Y, Kumano T, Ogino K. Intermediate representation for stiff virtual objects. In: Virtual reality annual international symposium, 1995. p. 203–10.
[2] Astley OR, Hayward V. Multirate haptic simulation achieved by coupling finite element meshes through norton equivalents. In: Proceedings of IEEE international conference on robotics and automation, 1998.
[3] Baraff D, Witkin AP. Large steps in cloth simulation. In: Proceedings of ACM SIGGRAPH, 1998.
[4] Barbagli F, Prattichizzo D, Salisbury KJ. Dynamic local models for stable multi-contact haptic interaction with deformable objects. In: Proceedings of haptics symposium, 2003.
[5] Barbič J, James DL. Real-time subspace integration for St. Venant-Kirchhoff deformable models. In: Proceedings of ACM SIGGRAPH, 2005.
[6] Barbič J, James DL. Six-DoF haptic rendering of contact between geometrically complex reduced deformable models. IEEE Transactions on Haptics 2008;1(1).
[7] Bouzit M, Burdea G, Popescu G, Boian R. The rutger master ii—new design force-feedback glove. IEEE Transactions on Mechatronics 2004;7(2).
[8] Çavuşoğlu MC, Tendick F. Multirate simulation for high fidelity haptic interaction with deformable objects in virtual environments. In: Proceedings of IEEE international conference on robotics and automation, 2000. p. 2458–65.
[9] Cobos S, Ferre M, Sanchez Uran M, Ortego J, Pena C. Efficient human hand kinematics for manipulation tasks. In: IEEE/RSJ international conference on intelligent robots and systems, 2008. p. 2246–51.
[10] Colgate JE, Brown JM. Factors affecting the z-width of a haptic display. In: IEEE international conference on robotics and automation, 1994. p. 3205–10.
[11] Colgate JE, Stanley MC, Brown JM. Issues in the haptic display of tool use. In: Proceedings of IEEE/RSJ international conference on intelligent robots and systems, 1995. p. 140–5.
[12] Constantinescu D, Salcudean SE, Croft EA. Local model of interaction for realistic manipulation of rigid virtual worlds. International Journal of Robotics Research 2005;24(10).
[13] de Pascale M, Sarcuni G, Prattichizzo D. Real-time soft-finger grasping of physically based quasi-rigid objects. In: Proceedings of the world haptics conference, Pisa, Italy, 18–20 March 2005. p. 545–6.
[14] Duriez C, Andriot C, Kheddar A. Signorini's contact model for deformable objects in haptic simulations. In: Proceedings of IEEE/RSJ IROS, 2004.
[15] Duriez C, Dubois F, Kheddar A, Andriot C. Realistic haptic rendering of interacting deformable objects in virtual environments. Proceedings of IEEE TVCG 2006;12(1).
[16] Garre C, Otaduy MA. Haptic rendering of complex deformations through handle-space force linearization. In: Proceedings of world haptics conference. IEEE Robotics and Automation Society, March 2009.
[17] Garre C, Otaduy MA. Toward haptic rendering of full-hand touch. In: Proceedings of Spanish computer graphics conference (CEIG), 2009.
[18] Gibson SF, Mirtich BV. A survey of deformable modeling in computer graphics. Technical Report, Mitsubishi Electric Research Laboratory, 1997.
[19] Golub GH, Van Loan CF. Matrix computations. 3rd ed. Johns Hopkins University Press; 1996.
[20] Harmon D, Vouga E, Tamstorf R, Grinspun E. Robust treatment of simultaneous collisions. In: Proceedings of ACM SIGGRAPH, 2008.
[21] Johnson DE, Willemsen P, Cohen E. 6-DOF haptic rendering using spatialized normal cone search. IEEE TVCG 2005;11(6).
[22] Kaufman DM, Sueda S, James DL, Pai DK. Staggered projections for frictional contact in multibody systems. In: Proceedings of ACM SIGGRAPH Asia, 2008.
[23] Kawasaki H, Mouri T, Osama M, Sugihashi Y, Ohtuka Y, Ikenohata S, et al. Development of five-fingered haptic interface: Hiro ii. In: Proceedings of ICAT, 2005.
[24] Kim YJ, Otaduy MA, Lin MC, Manocha D. Six-degree-of-freedom haptic rendering using incremental and localized computations. Presence 2003;12(3): 277–95.
[25] Klatzky RL, Lederman SJ. Touch. In: Experimental psychology. Weiner IB, Editor-in-Chief. Handbook of psychology, vol. 4. 2003. p. 147–76.
[26] Kuchenbecker KJ, Fiene J, Niemeyer G. Improving contact realism through event-based haptic feedback. IEEE TVCG 2006;12(2).
[27] Kurihara T, Miyata N. Modeling deformable human hands from medical images. In: Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation, 2004, p. 357–66.
[28] Lin MC, Manocha D. Collision and proximity queries. In: Goodman JE, O'Rourke J, editors. Handbook of discrete and computational geometry. CRC Press; 2003.
[29] Lin MC, Otaduy MA, editors. Haptic rendering: foundations, algorithms, and applications. AK Peters; 2008.
[30] Liu CK. Synthesis of interactive hand manipulation. In: Proceedings of ACM SIGGRAPH/Eurographics symposium on computer animation, 2008.
[31] McNeely W, Puterbaugh K, Troy J. Six degree-of-freedom haptic rendering using voxel sampling. In: Proceedings of ACM SIGGRAPH, 1999. p. 401–8.
[32] Mirtich B. Fast and accurate computation of polyhedral mass properties. Journal of Graphical Tools 1996;1(2).
[33] Monroy M, Oyarzabal M, Ferre M, Campos A, Barrio J. Masterfinger: multi-finger haptic interface for collaborative environments. In: Proceedings of Eurohaptics, 2008.
[34] Müller M, Gross M. Interactive virtual materials. In: Proceedings of graphics interface, 2004.
[35] Nealen A, Müller M, Keiser R, Boxermann E, Carlson M. Physically based deformable models in computer graphics. Eurographics STAR, 2005.
[36] Ortega M, Redon S, Coquillart S. A six-degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. IEEE TVCG 2007;13(3).
[37] Otaduy MA, Gross M. Transparent rendering of tool contact with compliant environments. In: Proceedings of world haptics conference, 2007.
[38] Otaduy MA, Lin MC. Sensation preserving simplification for haptic rendering. ACM transactions on graphics (proceedings of ACM SIGGRAPH), 2003. p. 543–53.
[39] Otaduy MA, Lin MC. A modular haptic rendering algorithm for stable and transparent 6-DoF manipulation. IEEE Transactions on Robotics 2006;22(4).
[40] Otaduy MA, Tamstorf R, Steinemann D, Gross M. Implicit contact handling for deformable objects. In: Proceedings of Eurographics, 2009.
[41] Picinbono G, Lombardo J-C, Delingette H, Ayache N. Anisotropic elasticity and force extrapolation to improve realism of surgery simulation. In: Proceedings of ICRA, 2000.
[42] Ruspini D, Kolarov K, Khatib O. The haptic display of complex graphical environments. In: Proceedings of ACM SIGGRAPH, 1997. p. 345–52.
[43] Saupin G, Duriez C, Cotin S. Contact model for haptic medical simulation. In: Proceedings of international symposium on computational models for biomedical simulation, 2008.
[44] Shabana AA. Dynamics of multibody systems. John Wiley and Sons; 1989.
[45] Spillman J, Becker M, Teschner M. Non-iterative computation of contact forces for deformable objects. Journal of WSCG 2007;15(1–3):33–40.
[46] Sueda S, Kaufman A, Pai DK. Musculotendon simulation for hand animation. In: ACM transactions on graphics (proceedings of ACM SIGGRAPH), vol. 27 (3), 2008.
[47] Teschner M, Kimmerle S, Heidelberger B, Zachmann G, Raghupathi L, Furhmann A, et al. Collision detection for deformable objects. Computer Graphics Forum 2005;24(1).
[48] Toledo S, Chen D, Rotkin V. Taucs: a library for sparse linear solvers, 2003.
[49] Zilles C, Salisbury K. A constraint-based god object method for haptics display. In: Proceedings of IEEE/RSJ international conference on intelligent robotics and systems, 1995.