Haptic Display of Interaction between Textured Models

Miguel A. Otaduy Nitin Jain Avneesh Sud Ming C. Lin

Department of Computer Science
University of North Carolina at Chapel Hill

http://gamma.cs.unc.edu/HTextures

Figure 1: **Haptic Display of Interaction between Textured Models.** *From left to right: (a) high-resolution textured hammer (433K polygons) and CAD part (658K polygons), (b) low-resolution models (518 & 720 polygons), (c) hammer texture with fine geometric detail.*

ABSTRACT

Surface texture is among the most salient haptic characteristics of objects; it can induce vibratory contact forces that lead to perception of roughness. In this paper, we present a new algorithm to display haptic texture information resulting from the interaction between two textured objects. We compute contact forces and torques using low-resolution geometric representations along with texture images that encode surface details. We also introduce a novel force model based on directional penetration depth and describe an efficient implementation on programmable graphics hardware that enables interactive haptic texture rendering of complex models. Our force model takes into account important factors identified by psychophysics studies and is able to haptically display interaction due to fine surface textures that previous algorithms do not capture.

Keywords: haptics, textures, graphics hardware

1 INTRODUCTION

Haptic rendering provides a unique, two-way communication between humans and interactive systems, enabling bi-directional interaction via tactile sensory cues. By harnessing the sense of touch, haptic display can further enhance a user's experience in a multimodal synthetic environment, providing a more natural and intuitive interface with the virtual world. A key area in haptics that has received increasing attention is the rendering of *surface texture*, i.e. fine geometric features on an object's surface. The intrinsic surface property of texture is among the most salient haptic characteristics of objects. It can be a compelling cue to object identity, and it can strongly influence forces during manipulation [16]. In medical applications with limited visual feedback, such as minimally-invasive or endoscopic surgery [24], and virtual prototyping applications of mechanical assembly and maintainability assessment [27], accurate haptic feedback of surface detail is a key factor for successful meticulous operations.

Most of the existing haptic rendering algorithms have focused primarily on force rendering of rigid or deformable flat polygonal models. This paper addresses the simulation of forces and torques due to interaction *between two textured objects*. Effective physically-based force models have been proposed to render the interaction between the tip (a point) of a haptic probe and a textured object [18, 10]. However, no technique is known to display both interaction forces and torques between two textured models. In fact, computation of texture-induced forces using full-resolution geometric representations of the objects and handling contacts at micro-geometric scale is computationally prohibitive.

Similar to graphical texture rendering [2], objects with high combinatorial complexity (i.e. with a high polygon count) can be described by coarse representations with their fine geometric detail stored in texture images, which we will refer to as *haptic textures* in this paper. Given this representation and a new force model that captures the effect of geometric surface details, we are able to haptically display intricate interaction between highly complex models using haptic textures instead of actual surface geometry.

Main Contributions: In this paper, we introduce a physically-based algorithm for incorporating texture effects to haptic display of interaction between two polygonal models. This algorithm enables, for the first time, interactive haptic display of forces and torques due to fine surface details. The main results of our paper are:

- A novel force model for haptic texture rendering, based on the gradient of directional penetration depth, that accounts for important factors identified by psychophysics studies;
- A fast algorithm for approximating directional penetration depth between textured objects;
- An efficient implementation on programmable graphics hardware that enables interactive haptic display of forces and torques between complex textured models;

IEEE Visualization 2004
October 10-15, Austin, Texas, USA
0-7803-8788-0/04/$20.00 ©2004 IEEE

- A new approach to haptically render complex interaction due to fine surface details using simplified representations of the original models and the corresponding haptic textures.

Our algorithm can be integrated in state-of-the-art haptic rendering algorithms to enhance the range of displayed stimuli. We have successfully tested and demonstrated our algorithm and implementation on several complex textured models. Some examples are shown in Fig. 1. Subjects were able to perceive roughness of various surface textures.

Organization: The rest of the paper is organized as follows. In Sec. 2 we discuss related work. Sec. 3 defines key terminology and describes several important concepts central to our force model. Sec. 4 presents the force computation model. Sec. 5 introduces a simple yet effective algorithm for approximating directional penetration depth and its parallel implementation on graphics processors. We then describe our results in Sec. 6. Finally, we discuss and analyze our approach in Sec. 7 and conclude with possible future research directions in Sec. 8.

2 Previous Work

In this section we briefly discuss related work on haptic rendering and penetration depth computations.

2.1 Six Degree-of-Freedom Haptics

Haptic display of forces and torques between two interacting objects is commonly known as 6 degree-of-freedom (DoF) haptics. In all approaches to 6-DoF haptics, collision detection is a dominant computational cost. The performance of collision detection algorithms depends on the size of the input models, which in turn depends on the sampling density of the models, both for polygonal representations [23, 15, 11] and for voxel-based representations [17, 27].

To be correctly represented, surfaces with high-frequency geometric texture detail require higher sampling densities, thereby increasing the cost of collision detection. As a result, haptic rendering of forces between textured objects becomes computationally infeasible to achieve, and new representations must be considered.

Otaduy and Lin [20] recently suggested multiresolution representations to minimize the computational impact of collision detection and to adaptively select the appropriate resolution at each contact location. However, their approach filters out high resolution geometric features, thus ignoring all texture effects.

2.2 Haptic Texture Rendering

Rendering and perception of textures has been one of the most active areas in haptics research. Please refer to [16] for a survey on psychophysics of tactile texture perception. Klatzky and Lederman made important distinctions between perception of textures with bare skin vs. perception through a rigid object. When perceived through a rigid probe, roughness of a textured surface is encoded as vibration.

Several researchers have successfully developed haptic texture rendering techniques for interaction between a probe point and an object, using coarse geometric approximations and geometric texture images. These techniques use the idea of computing geometry-dependent high frequency forces, which transmit vibratory information to the user, and are perceived as virtual roughness. Minsky [18] showed that texture information can be conveyed by displaying forces on the tangent plane defined by the contact normal. Minsky computed a texture-induced force proportional to the gradient of a 2D height field stored in a texture map. Ho et al. [10] have proposed techniques that alter the magnitude and direction of 3D normal force based on height field gradient. Siira and Pai [26] followed a stochastic approach, where texture forces are computed according to a Gaussian distribution.

All these techniques exploit the fact that, for point-object contact, a pair of texture coordinates can be well defined, and this is used to query height fields stored in texture maps. Note that only geometric effects of one object are captured. We are interested in rendering forces occurring during the interaction of two surfaces. In this case, the geometric interaction is not limited to and cannot be described by a pair of contact points. Moreover, the local kinematics of the contact between two surfaces include rotational degrees of freedom, not captured by point-based haptic rendering methods.

Choi and Tan [3] have studied the influence of collision detection and penetration depth computation on point-based haptic rendering, and their findings appear to be applicable to 6-DoF haptics as well.

2.3 Penetration Depth Computation

Several algorithms [12, 6, 5, 13] have been proposed for computing a measure of penetration depth using various definitions. However, each of them assumes that at least one of the input models is a convex polytope. It is commonly known that if two polytopes intersect, then the difference of their reference vectors with respect to the origin of the world coordinate system lies in their convolution or Minkowski sum [8]. The problem of penetration depth computation reduces to calculating the minimum distance from the origin to the boundary of the Minkowski sum of two polyhedra. The worst case complexity for two general, non-convex polyhedra can be as high as $O(m^3n^3)$, where m,n are the number of polygons in each model. Kim et al. [14] presented an algorithm for estimating penetration depth between two polyhedral models using rasterization hardware and hierarchical refinement. Although it offers better performance than previous techniques, this approach may take up to minutes to compute the penetration depth, making it inadequate for haptic simulation.

In this paper we present a new algorithm to estimate directional penetration depth between models described by low-resolution representations and haptic textures. Unlike the algorithm by Kim et al. [14], it does not compute the global penetration depth between two models, but its performance makes it suitable for haptic display.

3 Preliminaries

In this section we first introduce notation used in the paper. Then, we present definitions related to penetration depth, which is an essential element of our force model. Finally, we describe the computational pipeline for haptic rendering of interaction between textured models.

3.1 Notations

A *height field* H is defined as a set $H = \{(x,y,z) \mid z = h(x,y), (x,y,z) \in \mathbb{R}^3\}$. We call $h : \mathbb{R}^2 \to \mathbb{R}$ a *height function*. Let $\mathbf{q}$ denote a point in $\mathbb{R}^3$, let $\mathbf{q}_{xyz} = (q_x\ q_y\ q_z)^T$ denote the coordinates of $\mathbf{q}$ in a global reference system, and $\mathbf{q}_{uvn} = (q_u\ q_v\ q_n)^T$ its coordinates in a rotated reference system $\{\mathbf{u},\mathbf{v},\mathbf{n}\}$. A surface patch $S \subset \mathbb{R}^3$ can be represented as a height field along a direction $\mathbf{n}$ if $q_n = h(q_u,q_v), \forall \mathbf{q} \in S$. Then, we can define a mapping $g : D \to S, D \subset \mathbb{R}^2$, as $g(q_u,q_v) = \mathbf{q}_{xyz}$, where:

$$h(q_u,q_v) = q_n = \mathbf{n} \cdot \mathbf{q}_{xyz} = \mathbf{n} \cdot g(q_u,q_v) \tag{1}$$

The inverse of the mapping g is the orthographic projection of S onto the plane $(\mathbf{u},\mathbf{v})$ along the direction $\mathbf{n}$.

3.2 Definitions of Penetration Depth

Penetration depth δ between two intersecting polytopes is typically defined as the minimum translational distance required to separate them (see Fig. 2-b). As mentioned in Sec. 2.3, this distance is equivalent to the distance from the origin to the Minkowski sum of the polyhedra. *Directional penetration depth* $\delta_{\mathbf{n}}$ along the direction $\mathbf{n}$ is defined as the minimum translation along $\mathbf{n}$ to separate the polyhedra (see Fig. 2-c). The penetration depth between two intersecting surface patches will be referred to as *local penetration depth*.

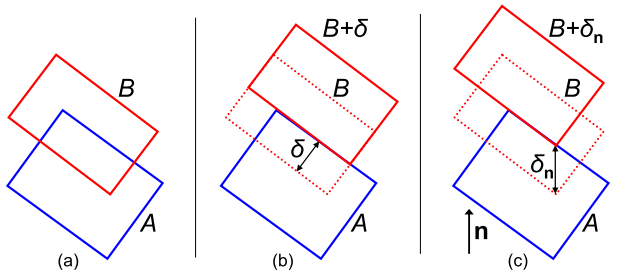

Figure 2: **Definitions of Penetration Depth.** *(a) Intersecting objects A and B, (b) global penetration depth δ, and (c) directional penetration depth δ_n along* $\mathbf{n}$.

Let us assume that two intersecting surface patches S_A and S_B can be represented as height fields along a direction $\mathbf{n}$. Consequently, S_A and S_B can be parameterized by orthographic projection along $\mathbf{n}$, as expressed in Sec. 3.1. As a result of the parameterization, we obtain mappings $g_A : D_A \rightarrow S_A$ and $g_B : D_B \rightarrow S_B$, as well as height functions $h_A : D_A \rightarrow \mathbb{R}$ and $h_B : D_B \rightarrow \mathbb{R}$. The directional penetration depth $\delta_{\mathbf{n}}$ of the surface patches S_A and S_B is the maximum height difference along the direction $\mathbf{n}$, as illustrated in Fig. 3 by a 2D example. Therefore, we can define the directional penetration depth $\delta_{\mathbf{n}}$ as:

$$\delta_{\mathbf{n}} = \max_{(u,v)\in(D_A\cap D_B)} \big(h_A(u,v) - h_B(u,v)\big) \qquad (2)$$

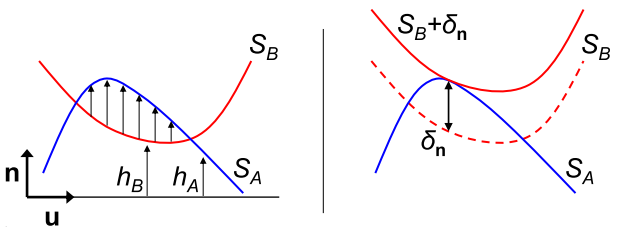

Figure 3: **Penetration Depth of Height Fields.** *Directional penetration depth of surface patches expressed as height difference.*

3.3 Haptic Display Pipeline

We assume that the interacting objects can be described as parameterized low-resolution triangle meshes with texture maps that store fine geometric detail. In a haptic simulation of object-object interaction, the object whose motion is controlled by the user is called the *probe object*. Contacts between the probe object and the rest of the objects in the environment generate forces that are displayed to the user.

Following a common approach in 6-DoF haptics, we simulate the dynamics of the probe object as a result of contact forces and a virtual coupling force that ensures stable interaction with the user [1]. We propose a novel algorithm for computing contact forces, taking into account texture effects. We follow the steps below to compute contact forces:

1. Each haptic simulation frame starts by performing collision detection between the low-resolution meshes. We then identify intersecting surface patches as contacts. We characterize each contact by a pair of contact points on the patches and a penetration direction $\mathbf{n}$.

2. For each contact, we compute force and torque using our novel force model for texture rendering, based on the penetration depth and its gradient. The penetration depth is approximated taking into account fine geometric detail stored in haptic textures.

3. The forces and torques of all contacts are combined to compute the net force and torque on the probe object.

Other effects, such as friction [9], can easily be incorporated into this display pipeline using the contact information computed between the low-resolution meshes.

4 A Force Model for Texture Rendering

In this section we describe our force model for haptic display of interaction between textured surfaces. We first show how factors highlighted by psychophysics studies are taken into account. Then, we introduce a penalty-based force model for texture rendering. Finally, we present the formulation of the gradient of penetration depth used in our force model.

4.1 Foundation of the Proposed Force Model

Roughness of surface textures perceived through a rigid probe is mainly encoded as vibration and strongly influences the forces that must be applied to manipulate the objects [16]. In point-based haptic texture rendering, vibrating forces are commonly computed using a height field gradient [18, 10]. Our force model generalizes the point-based approach by computing forces based on the *gradient of penetration depth* between two objects.

Based on psychophysics studies, Klatzky and Lederman [16] highlight factors influencing perception of roughness through a rigid spherical probe. These factors are:

Probe Radius: For spherical probes, the texture frequency at which perception of roughness is maximum depends on probe radius. At low frequencies, roughness increases with texture frequency, but after reaching a peak, roughness decreases as texture frequency increases. Our conjecture is that roughness perception is tightly coupled to the trajectory traced by the probe, which can be regarded as an offset surface of the perceived geometry. Okamura and Cutkosky [19] also modeled interaction between robotic fingers and textured surfaces by tracing offset surfaces. They defined an offset surface as the boundary of the Minkowski sum of a given surface and a sphere. Therefore, the height of the offset surface at a particular point is the distance to the boundary of the Minkowski sum for a particular position of the probe, also known to be the penetration depth[1]. In other words, the height of the offset surface reflects the distance that the probe must move in order to avoid interpenetration with the surface. Since, for spherical probes, perception of roughness seems to be tightly coupled with the oscillation of offset surfaces, in our force model for general surfaces we have taken into account the *variation of penetration depth*, i.e. its gradient.

Normal Force: Perception of roughness grows monotonically with normal force. This relation is also captured by our force model in

[1] Actually, the height of the offset surface is the distance to the surface along a particular direction, so the distance to the boundary of the Minkowski sum must also be measured along a particular direction. This is known to be the *directional penetration depth*.

a qualitative way, in making tangential forces and torques proportional to the normal force.

Exploratory Speed: The exploratory speed, or velocity of the probe in the plane of contact with the surface, affects the perception of roughness. Our force model is intrinsically geometry-based, but in a haptic simulation dynamic effects are introduced by the haptic device and the user. We have analyzed the dynamic behavior of our force model, and we have observed that vibratory motion produced by simulated forces behaves in a way similar to physical roughness perception. The results of our experiments are described in detail in [21].

The influence of probe geometry, normal force and exploratory speed is taken into consideration in the design of our force model, which will be presented next.

4.2 Penalty-Based Texture Force

For two objects A and B in contact, we define a penalty-based force proportional to the penetration depth δ between them. Penalty-based forces are conservative, and they define an elastic potential field. In our force model we have extended this principle to compute texture-induced forces between two objects.

We define an elastic penetration energy U with stiffness k as:

$$U = \frac{1}{2} k \delta^2 \tag{3}$$

Based on this energy, we define force $\mathbf{F}$ and torque $\mathbf{T}$ as:

$$\begin{pmatrix} \mathbf{F} \\ \mathbf{T} \end{pmatrix} = -\nabla U = -k\delta\,(\nabla\delta) \tag{4}$$

where $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}, \frac{\partial}{\partial \theta_x}, \frac{\partial}{\partial \theta_y}, \frac{\partial}{\partial \theta_z}\right)$ is the gradient in 6-DoF configuration space.

As described in Sec. 3.3, each contact between objects A and B can be described by a pair of contact points $\mathbf{p_A}$ and $\mathbf{p_B}$, and by a penetration direction $\mathbf{n}$. We assume that, locally, the penetration depth between objects A and B can be approximated by the directional penetration depth $\delta_\mathbf{n}$ along $\mathbf{n}$. We rewrite Eq. 4 for $\delta_\mathbf{n}$ in a reference system $\{\mathbf{u},\mathbf{v},\mathbf{n}\}$[2]. In this case, Eq. 4 reduces to:

$$\begin{pmatrix} F_u & F_v & F_n & T_u & T_v & T_n \end{pmatrix}^T = -k\delta_\mathbf{n} \begin{pmatrix} \frac{\partial \delta_\mathbf{n}}{\partial u} & \frac{\partial \delta_\mathbf{n}}{\partial v} & 1 & \frac{\partial \delta_\mathbf{n}}{\partial \theta_u} & \frac{\partial \delta_\mathbf{n}}{\partial \theta_v} & \frac{\partial \delta_\mathbf{n}}{\partial \theta_n} \end{pmatrix}^T \tag{5}$$

where θ_u, θ_v and θ_n are the rotation angles around the axes $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{n}$ respectively.

The force and torque on object A (and similarly on object B) for each contact can be expressed in the global reference system as:

$$\begin{aligned} \mathbf{F_A} &= (\mathbf{u}\ \mathbf{v}\ \mathbf{n})\,(F_u\ F_v\ F_n)^T \\ \mathbf{T_A} &= (\mathbf{u}\ \mathbf{v}\ \mathbf{n})\,(T_u\ T_v\ T_n)^T \end{aligned} \tag{6}$$

As explained in Sec. 3.3, forces and torques of all contacts are summed up to compute the net force and torque.

Generalizing Minsky's approach [18], we define tangential forces F_u and F_v proportional to the gradient of penetration depth. However, we also define a penalty-based normal force and gradient-dependent torque that describe full 3D object-object interaction. In addition, in our model the tangential force and the torque are proportional to the normal force, which is consistent with psychophysics studies showing that perceived roughness increases with the magnitude of the normal force [16].

[2] $\mathbf{u}$ and $\mathbf{v}$ may be selected arbitrarily as long as they form an orthonormal basis with $\mathbf{n}$.

4.3 Penetration Depth and Gradient

In our formulation, δ and $\delta_\mathbf{n}$ are functions defined on a 6-DoF configuration space. We have opted for central differencing over one-sided differencing to approximate $\nabla\delta_\mathbf{n}$, because it offers better interpolation properties and higher order approximation. The partial derivatives are computed as:

$$\frac{\partial \delta_\mathbf{n}}{\partial u} = \frac{\delta_\mathbf{n}(u+\Delta u, v, n, \theta_u, \theta_v, \theta_n) - \delta_\mathbf{n}(u-\Delta u, v, n, \theta_u, \theta_v, \theta_n)}{2\Delta u} \tag{7}$$

and similarly for $\frac{\partial \delta_\mathbf{n}}{\partial v}$, $\frac{\partial \delta_\mathbf{n}}{\partial \theta_u}$, $\frac{\partial \delta_\mathbf{n}}{\partial \theta_v}$ and $\frac{\partial \delta_\mathbf{n}}{\partial \theta_n}$.

$\delta_\mathbf{n}(u+\Delta u, ...)$ can be obtained by translating object A a distance Δu along the $\mathbf{u}$ axis and computing the directional penetration depth. A similar procedure is followed for other penetration depth values.

5 Directional Penetration Depth

In this section we present an algorithm for approximating local directional penetration depth for textured models and describe a parallel implementation on graphics hardware.

5.1 Approximate Directional Penetration Depth between Textured Models

A contact between objects A and B is defined by two intersecting surface patches S_A and S_B. The surface patch S_A is approximated by a low-resolution surface patch $\hat{S}_A$ (and similarly for S_B). We define $f_A : \hat{S}_A \rightarrow S_A$, a mapping function from the low-resolution surface patch $\hat{S}_A$ to the surface patch S_A.

Collision detection between two low-resolution surfaces patches $\hat{S}_A$ and $\hat{S}_B$ returns a penetration direction $\mathbf{n}$. Let us assume that both S_A and $\hat{S}_A$ (and similarly for S_B and $\hat{S}_B$) can be represented as height fields along $\mathbf{n}$, following the definition in Sec. 3.1. Given a rotated reference system $\{\mathbf{u},\mathbf{v},\mathbf{n}\}$, we can project S_A and $\hat{S}_A$ orthographically along $\mathbf{n}$ onto the plane $(\mathbf{u},\mathbf{v})$. As the result of this projection, we obtain mappings $g_A : D_A \rightarrow S_A$ and $\hat{g}_A : \hat{D}_A \rightarrow \hat{S}_A$. We define $\bar{D}_A = D_A \cap \hat{D}_A$.

The mapping function g_A can be approximated by a composite mapping function $f_A \circ \hat{g}_A : \bar{D}_A \rightarrow S_A$ (See Fig. 4). From Eq. 1, we define an approximate height function $\hat{h} : \bar{D}_A \rightarrow \mathbb{R}$ as:

$$\hat{h}(u,v) = \mathbf{n} \cdot (f_A \circ \hat{g}_A(u,v)) \tag{8}$$

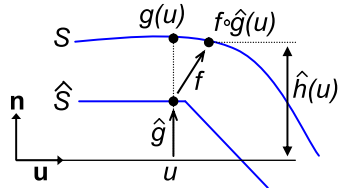

Figure 4: **Approximate Height Function.** *Height function of a surface patch approximated by a composite mapping function.*

Given approximate height functions $\hat{h}_A$ and $\hat{h}_B$, a domain $D = \bar{D}_A \cap \bar{D}_B$, and Eq. 2, we can approximate the directional penetration depth $\delta_\mathbf{n}$ of S_A and S_B by:

$$\hat{\delta}_\mathbf{n} = \max_{(u,v)\in D} \left(\hat{h}_A(u,v) - \hat{h}_B(u,v)\right) \tag{9}$$

Although this algorithm can be realized on CPUs, it is best suited for implementation on graphics processors (GPUs), as we will present next.

5.2 Computation on Graphics Hardware

As shown in Eq. 5, computation of 3D texture-induced force and torque according to our model requires the computation of directional penetration depth $\delta_{\mathbf{n}}$ and its gradient at every contact. From Eq. 7, this reduces to computing $\delta_{\mathbf{n}}$ all together at 11 configurations of object A[3]. As pointed out in section 2.3, computation of penetration depth using exact object-space or configuration-space algorithms is too expensive for haptic rendering applications. Instead, the approximation $\hat{\delta}_{\mathbf{n}}$ according to Eqs. 8 and 9 leads to a natural and efficient image-based implementation on programmable graphics hardware. The mappings $\hat{g}$ and f correspond, respectively, to orthographic projection and texture mapping operations, which are best suited for the parallel and grid-based nature of GPUs.

For every contact, we first compute $\hat{h}_B$, and then perform two operations for each of the 11 object configurations: compute $\hat{h}_A$ for the transformed object A, and then find the penetration depth $\hat{\delta}_{\mathbf{n}} = \max(\Delta\hat{h}) = \max\left(\hat{h}_A - \hat{h}_B\right)$[4].

Height Computation

In our GPU-based implementation, the mapping $f : \hat{S} \rightarrow S$ is implemented as a texture map that stores geometric detail of the high-resolution surface patch S. We refer to f as a "haptic texture". The mapping $\hat{g}$ is implemented by rendering $\hat{S}$ using an orthographic projection along $\mathbf{n}$. We compute the height function $\hat{h}$ in a fragment program. We obtain a point in S by looking up the haptic texture f and then we project it onto $\mathbf{n}$. The result is stored in a floating point texture t.

We choose geometric texture mapping over other methods for approximating h (e.g. rendering S directly or performing displacement mapping) in order to maximize performance. We store the input haptic texture f as a floating point texture, thus alleviating precision problems.

Max Search

The *max* function in Eq. 9 can be implemented as a combination of frame buffer read-back and CPU-based search. However, we avoid expensive read-backs by posing the *max* function as a binary search on the GPU [7]. Given two height functions $\hat{h}_A$ and $\hat{h}_B$ stored in textures t_1 and t_2, we compute their difference and store it in the depth buffer. We scale and offset the height difference to fit in the depth range. Height subtraction and copy to depth buffer are performed in a fragment program, by rendering a quad that covers the entire buffer. For a depth buffer with N bits of precision, the search domain is the integer interval $[0, 2^N)$. The binary search starts by querying if there is any value larger than 2^{N-1}. We render a quad at depth 2^{N-1} and perform an occlusion query [5], which will report if any pixel passed the depth test, i.e. the stored depth was larger than 2^{N-1}. Based on the result, we set the depth of a new quad and continue the binary search.

Gradient Computation

The height functions $\hat{h}_A(\pm\Delta u)$, $\hat{h}_A(\pm\Delta v)$ and $\hat{h}_A(\pm\Delta\theta_n)$ may be obtained by simply translating or rotating $\hat{h}_A(0)$. As a result, only 6 height functions $\hat{h}_A(0)$, $\hat{h}_B(0)$, $\hat{h}_A(\pm\Delta\theta_u)$ and $\hat{h}_A(\pm\Delta\theta_v)$ need to be computed for each pair of contact patches. These 6 height functions are tiled in one single texture t to minimize context switches and increase performance (See Fig. 5). Moreover, the domain of each height function is split into 4 quarters, each of which is mapped to one of the RGBA channels. This optimization allows us to exploit vector computation capabilities of fragment processors. As shown in Fig. 5, we also tile 11 height differences per contact in the depth buffer.

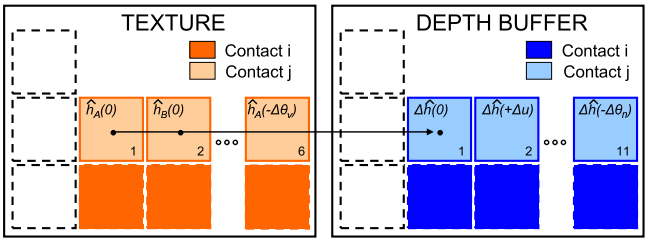

Figure 5: **Tiling in the GPU.** *Tiling of multiple height functions and contacts to minimize context switches between target buffers.*

Multiple Simultaneous Contacts:

The computational cost of haptic texture rendering increases linearly with the number of contact patches between the interacting objects. However, performance can be further optimized. In order to limit context switches, we tile the height functions associated with multiple pairs of contact patches in one single texture t, and we tile the height differences in the depth buffer, as shown in Fig. 5. We also minimize the cost of "max search" operations by performing occlusion queries on all contacts in parallel.

6 Results

We now describe the implementation details and results obtained with our haptic texture rendering algorithm, both in terms of force and motion characteristics, as well as performance.

6.1 Implementation Details

Our haptic texture rendering algorithm requires a preprocessing step. Input models are assumed to be 2-manifold triangle meshes with fine geometric details. We parameterize the meshes and create texture atlas storing surface positions. We also simplify the meshes to produce coarse resolution approximations which are used by the collision detection module. The parameterization must be preserved during the simplification process, and distortion must be minimized. Our implementation of parameterization-preserving simplification is based on existing approaches [25, 4].

As described in Sec. 3.3, before computing forces we perform collision detection between coarse-resolution models. We adapt the approach of Kim et al. [15] and decompose the models in convex pieces. Object interpenetration is considered to occur when objects are closer than a distance tolerance. In practice, by using this technique, penetration depth between the coarse resolution models is computed less frequently, thus accelerating collision detection.

For texture force computation, we compute each value of penetration depth between contact patches on a 50×50, 16-bit depth buffer. This resolution proved to be sufficient based on the results.

In our experiments we have used a 6-DoF *Phantom*^{TM} haptic device, a dual Pentium-4 2.4GHz processor PC with 2.0 GB of memory and an NVidia GeForce FX5950 graphics card, and Windows2000 OS. The penetration depth computation on graphics hardware is implemented using OpenGL plus OpenGL's ARB_fragment_program and GL_NV_occlusion_query extensions. Our haptic texture rendering cannot be stalled by the visual display of the scene; hence, it requires a dedicated graphics card. We display the full resolution scene on a separate commodity PC. The force update of the haptic device takes place at a frequency of 1kHz, but the haptic simulation is executed in a separate thread that updates the force input to a stabilizing virtual coupling [1] asynchronously.

[3] Note that, since we use central differencing to compute partial derivatives of $\delta_{\mathbf{n}}$, we need to transform object A to two different configurations and recompute $\delta_{\mathbf{n}}$. All together we compute $\delta_{\mathbf{n}}$ itself and 5 partial derivatives, hence 11 configurations

[4] We denote the height difference at the actual object configuration by $\Delta\hat{h}(0)$, and the height differences at the transformed configurations by $\Delta\hat{h}(\pm\Delta u)$, $\Delta\hat{h}(\pm\Delta v)$, $\Delta\hat{h}(\pm\Delta\theta_u)$, $\Delta\hat{h}(\pm\Delta\theta_v)$ and $\Delta\hat{h}(\pm\Delta\theta_n)$.

[5] http://www.nvidia.com/dev_content/nvopenglspecs/GL_NV_occlusion_query.txt

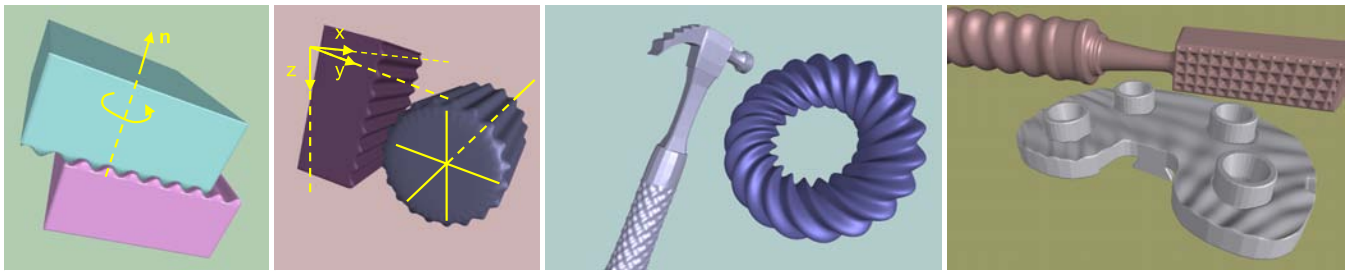

Figure 6: **Benchmark Models.** *From left to right: (a) textured blocks, (b) block and gear, (c) hammer and torus, (d) file and CAD part.*

6.2 Benchmarks

In our experiments we have used the models shown in Fig. 6. The complexity of full resolution textured models and their coarse resolution approximations is listed in Table 1.

Models	Full Res. Tris	Low Res. Tris	Low Res. Pcs
Block	65536	16	1
Gear	25600	1600	1
Hammer	433152	518	210
CAD Part	658432	720	390
File	285824	632	113
Torus	128000	532	114

Table 1: **Complexity of Benchmark Models.** *Number of triangles at full resolution (Full Res. Tris) and low resolution (Low Res. Tris), and number of convex pieces at low resolution (Low Res. Pcs).*

Notice the drastic simplification of the low resolution models. At this level all texture information is eliminated from the geometry, but it is stored in 1024×1024-size floating point textures. The number of convex pieces at coarse resolution reflects the geometric complexity for the collision detection module. Also notice that the *block* and *gear* models are fully convex at coarse resolution. The interaction between these models is described by one single contact, so they are better suited for analyzing force and motion characteristics in the simulations.

6.3 Conveyance of Roughness

We have performed experiments to test the conveyance of roughness with our haptic texture rendering algorithm.

Roughness under Translation: The gear and block models present ridges that interlock with each other. One of our experiments consisted of translating the block in the 3 Cartesian axes, while being in contact with the fixed gear, as depicted in Fig. 6-b. Fig. 7 shows the position of the block and the force exerted on it during 1500 frames of interactive simulation (approx. 3 seconds).

Notice that the force in the x direction, which is parallel to the ridges, is almost 0. Our model successfully yields this expected result, because the derivative of the penetration depth is 0 along the x direction. Notice also the staircase shape of the motion in the z direction, which reflects how the block rests for short periods of time on the ridges of the gear. The motion in the y direction resembles a staircase as well, but with small overshoots. These reflect the state between two successive interlocking situations, when the ridges are opposing each other. The wide frequency spectrum of staircase motion is possible due to the fine spatial resolution of penetration depth and gradient computation. Last, the forces in y and z are correlated with the motion profiles.

Roughness under Rotation: We placed two identical striped blocks interlocking each other, as shown in Fig. 6-a. We then performed small rotations of the upper block around the direction **n**, and observed the induced translation along that same direction. Fig. 8 shows the rotation and translation captured during 6000

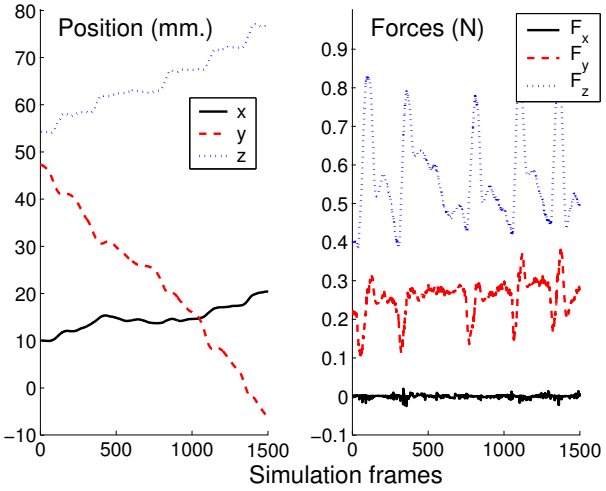

Figure 7: **Roughness under Translation.** *Position and force profiles generated while translating the model of a textured block in contact with a gear model, as shown in Fig. 6-b. Notice the staircase like motion in z, and the correlation between force and position changes.*

frames of interactive haptic simulation (approx. 12 seconds). Notice how the top block rises along **n** as soon as we rotate it slightly, thus producing a motion very similar to the one that occurs in reality. Point-based haptic rendering methods are unable to capture this type of effect. Our force model successfully produces the desired effect by taking into account the local penetration depth between the blocks. Also, the derivative of the penetration depth produces a physically-based torque in the direction **n** that opposes the rotation.

Summary of Experiments: From the experiments described above, we conclude that our force model successfully captures roughness properties of objects with fine geometric detail. We have also conducted informal experiments where subjects were asked to explore a textured plate with a virtual probe, while only the untextured coarse-resolution models were displayed graphically on the screen. Hence, the subjects could only recognize the texture patterns through haptic cues. The reported experiences were promising, as subjects were able to successfully describe regular patterns such as stripes, but had more difficulty with irregular patterns. This result is what we expect when real, physical textured models are explored.

6.4 Performance Tests

One of the key issues to achieve realistic haptic rendering is very high force update rate. High update rates enhance the stability of the system, as well as the range of stimuli that can be displayed. We have tested the performance of our haptic texture rendering algorithm and its implementation in scenarios where the coarse resolution models present complex contact configurations. These scenarios consist of a file scraping a rough CAD part, and a textured

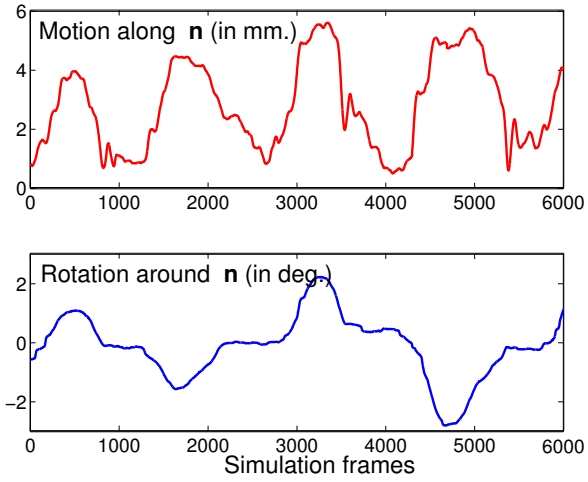

Figure 8: **Roughness under Rotation.** *Motion profile obtained by rotating one textured block on top of another one, as depicted in Fig. 6-a. Notice the translation induced by the interaction of ridges during the rotational motion.*

hammer touching a wrinkled torus. In particular, we show timings for 500 frames of the simulation of the file interacting with the CAD part in Fig. 9. The graph reflects the time spent on collision detection between the coarse-resolution models (an average of 2ms), the time spent on haptic texture rendering, and the total time per frame, which is approximately equal to the sum of the previous two. In this experiment we computed each value of penetration depth on a 50×50 16-bit depth buffer (See Sec. 5.2). As shown in Sec. 6.3, this proved to be sufficient to display convincing roughness stimuli.

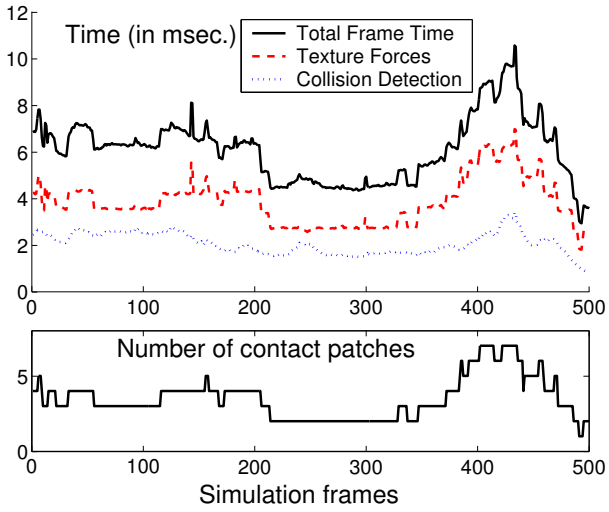

Figure 9: **Timings.** *Performance analysis and number of clustered contact patches during* 500 *simulation frames of a file model scraping a CAD part, as shown in Fig. 6-d. In this complex contact scenario we are able to maintain a haptic frame rate between 100Hz and 200Hz.*

In this particularly challenging experiment we were able to obtain haptic update rates between 100Hz and 200Hz. The dominant cost appears to be the haptic texture rendering, which depends nearly linearly on the number of contacts. The achieved force update rate may not be high enough to render textures with very high spatial frequency. However, as shown in Sec. 6.3, our proposed force model enables perception of roughness stimuli that were not captured previously by earlier methods. Moreover, in Fig. 9 we show performance results for a contact configuration in which large areas of the file at many different locations are in close proximity with the CAD part. In fact, collision detection using coarse-resolution models reports an average of 104 pairs of convex pieces in close proximity, which are later clustered into as many as 7 contacts. Using the full-resolution models, the number of contact pairs in close proximity would increase by several orders of magnitude, and simply handling collision detection would become challenging at the desired haptic rendering frame rates. Furthermore, as the support for programming on GPUs and capabilities of GPUs continue to grow at a rate faster than Moore's Law, we expect the performance of our algorithm to reach KHz update rates in the near future.

7 Discussion and Analysis

In Sec. 6.3 we have analyzed forces and motion generated by our algorithm during actual haptic simulations. We have further analyzed the properties of the force model presented in Sec. 4 by simulating its behavior in experiments similar to the ones conducted in psychophysics studies [16]. Our main conclusion is that the acceleration produced by our force model matches qualitatively the behavior of roughness perception as a function of texture frequency. A detailed description of the experiments we have conducted can be found in [21].

Our force model and implementation present a few limitations, some of which are common to existing haptic rendering methods. Next we discuss these limitations.

7.1 Force Model

In some contact scenarios with large contact areas, the definition of a local and directional penetration depth is not applicable. An example is the problem of screw insertion. Situations also exist in which local geometry cannot be represented as height fields, and the gradient of directional penetration depth may not capture the effect of interlocking features.

As shown in Sec. 6, in practice our force model generates forces that create a realistic perception of roughness for object-object interaction; however, one essential limitation of penalty-based methods and impedance-type haptic devices is the inability to enforce motion constraints. Our force model attempts to do so by increasing tangential contact stiffness when the gradient of penetration depth is high. Implicit integration of the motion of the probe object allows for high stiffness and, therefore, small interpenetrations, but the perceived stiffness of rigid contact is limited through virtual coupling for stability reasons. New constraint-based haptic rendering techniques and perhaps other haptic devices [22] will be required to properly enforce constraints.

A very important issue in every force model for haptic rendering is its stability. Choi and Tan [3] have shown that even passive force models may suffer from a problem called *aliveness*. In our algorithm, discontinuities in the collision detection between low-resolution models are possible sources of aliveness.

7.2 Frequency and Sampling Issues

As with other sample-based techniques, our haptic texture rendering algorithm is susceptible to aliasing problems. Here we discuss different aliasing sources and suggest some solutions.

Input textures: The resolution of input textures must be high enough to capture the highest spatial frequency of input models, although input textures can be filtered as a preprocessing step to downsample and reduce their size.

Image-based computation: In the height function computation step, buffer resolution must be selected so as to capture the spatial

frequency of input models. Buffer size, however, has a significant impact in the performance of force computation.

Discrete derivatives: Penetration depth may not be a smooth function. This property results in an infinitely wide frequency spectrum, which introduces aliasing when sampled. Differentiation aggravates the problem, because it amplifies higher frequencies. The immediate consequence in our texture rendering approach is that the input texture frequencies have to be low enough so as to represent faithfully their derivatives. This limitation is common to existing point-based haptic rendering methods [18] as well.

Temporal sampling. Force computation undergoes temporal sampling too. The Nyquist rate depends on object speed and spatial texture frequency. Image-based filtering prior to computation of penetration depth may remove undesirable high frequencies, but it may also remove low frequencies that would otherwise appear due to the nonlinearity of the max search operation. In other words, filtering a texture with very high frequency may incorrectly remove all torque and tangential forces. Temporal supersampling appears to be a solution to the problem, but is often infeasible due to the high update rates required by haptic simulation.

8 Conclusion

We have introduced a new haptic rendering algorithm for displaying interaction between two textured models, based on localized directional penetration depth and its gradient. We have also presented an image-based implementation on programmable graphics hardware that enables interactive haptic display between complex textured models for the first time. We have further shown that, using a coarse geometric representation with haptic textures that encode fine surface details, it is possible to render contact forces and torques between two interacting textured models at haptic rates.

Several possible directions for future research remain, including but not limited to:

- Interactive haptic texture synthesis;
- Addition of constraint forces for fine motion and dexterous manipulation;
- Further analysis of human factors.

Finally, we would like to integrate our haptic rendering system with different applications, such as assisted technology, surgical training, and virtual prototyping.

Acknowledgments

This research is supported in part by a fellowship of the Government of the Basque Country, National Science Foundation, Office of Naval Research, U.S. Army Research Office, and Intel Corporation. We would like to thank Roberta Klatzky, Susan Lederman, Dinesh Manocha, Russ Taylor, Fred Brooks, Mark Foskey, Bill Baxter, the UNC Gamma group and the anonymous reviewers for their feedback on the earlier drafts of this paper.

References

[1] R. J. Adams and B. Hannaford. A two-port framework for the design of unconditionally stable haptic interfaces. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1998.

[2] Edwin E. Catmull. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. PhD thesis, Dept. of CS, U. of Utah, December 1974.

[3] S. Choi and H. Z. Tan. Aliveness: Perceived instability from a passive haptic texture rendering system. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[4] J. Cohen, M. Olano, and D. Manocha. Appearance preserving simplification. In *Proc. of ACM SIGGRAPH*, pages 115–122, 1998.

[5] D. Dobkin, J. Hershberger, D. Kirkpatrick, and S. Suri. Computing the intersection-depth of polyhedra. *Algorithmica*, 9:518–533, 1993.

[6] E.G. Gilbert and C.J. Ong. New distances for the separation and penetration of objects. In *Proceedings of International Conference on Robotics and Automation*, pages 579–586, 1994.

[7] N. K. Govindaraju, B. Lloyd, W. Wang, M. C. Lin, and D. Manocha. Fast computation of database operations using graphics processors. *Proc. of ACM SIGMOD*, 2004.

[8] L. J. Guibas and J. Stolfi. Ruler, compass and computer: the design and analysis of geometric algorithms. In R. A. Earnshaw, editor, *Theoretical Foundations of Computer Graphics and CAD*, volume 40 of *NATO ASI Series F*, pages 111–165. Springer-Verlag, 1988.

[9] V. Hayward and B Armstrong. A new computational model of friction applied to haptic rendering. *Experimental Robotics VI*, 2000.

[10] C.-H. Ho, C. Basdogan, and M. A. Srinivasan. Efficient point-based rendering techniques for haptic display of virtual objects. *Presence*, 8(5):pp. 477–491, 1999.

[11] D. Johnson and P. Willemsen. Six degree of freedom haptic rendering of complex polygonal models. In *Proc. of Haptics Symposium*, 2003.

[12] S. S. Keerthi and K. Sridharan. Efficient algorithms for computing two measures of depth of collision between convex polygons. Technical Report, 1989.

[13] Y. Kim, M. Lin, and D. Manocha. DEEP: an incremental algorithm for penetration depth computation between convex polytopes. *Proc. of IEEE Conference on Robotics and Automation*, pages 921–926, 2002.

[14] Y. Kim, M. Otaduy, M. Lin, and D. Manocha. Fast penetration depth computation for physically-based animation. *Proc. of ACM Symposium on Computer Animation*, 2002.

[15] Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha. Six-degree-of-freedom haptic rendering using incremental and localized computations. *Presence*, 12(3):277–295, 2003.

[16] R. L. Klatzky and S. J. Lederman. Perceiving texture through a probe. In M. L. McLaughlin, J. P. Hespanha, and G. S. Sukhatme, editors, *Touch in Virtual Environments*, chapter 10, pages 180–193. Prentice Hall PTR, Upper Saddle River, NJ, 2002.

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

[18] M. Minsky. *Computational Haptics: The Sandpaper System for Synthesizing Texture for a Force-Feedback Display*. PhD thesis, Ph.D. Dissertation, Program in Media Arts and Sciences, MIT, 1995. Thesis work done at UNC-CH Computer Science.

[19] A. M. Okamura and M. R. Cutkosky. Feature detection for haptic exploration with robotic fingers. *International Journal of Robotics Research*, 20(12):925–938, 2001.

[20] M. A. Otaduy and M. C. Lin. Sensation preserving simplification for haptic rendering. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)*, pages 543–553, 2003.

[21] M. A. Otaduy and M. C. Lin. A perceptually-inspired force model for haptic texture rendering. *Proc. of Symposium APGV*, 2004.

[22] M. Peshkin and J. E. Colgate. Cobots. *Industrial Robot*, 26(5):335–341, 1999.

[23] D. Ruspini and O. Khatib. A framework for multi-contact multi-body dynamic simulation and haptic display. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[24] J. K. Salisbury. Making graphics physically tangible. *Communications of the ACM*, 42(8), 1999.

[25] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. *Proc. of ACM SIGGRAPH*, pages 409–416, 2001.

[26] J. Siira and D. K. Pai. Haptic textures – a stochastic approach. *Proc. of IEEE International Conference on Robotics and Automation*, pages 557–562, 1996.

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