Haptic Editing of Decoration and Material Properties

Laehyun Kim Gaurav S. Sukhatme Mathieu Desbrun

University of Southern California,
{laehyunk, gaurav, desbrun}@usc.edu

Abstract

In this paper, we explore haptic editing as a powerful way to enhance haptic realism. Building upon the framework of a previous implicit-based haptic technique, our haptic decoration technique allows the user to first paint directly on 3D models, and then to sense the thickness variation due to the added paint. Meanwhile, material editing permits the user to edit and feel local material properties such as friction and stiffness in a natural way. In addition, we extended the initial haptic model to support some novel features including a magnetic surface attraction that forces the tool tip to stick to the surface of complex models.

1 Introduction

A haptic interface allows the user to touch, explore, paint, and manipulate 3d models in an intuitive way thanks to realistic haptic feedback. In order to accomplish haptical realism, many haptic rendering algorithms [19, 30, 25, 24, 11, 2, 26, 28, 15] have been suggested over the last few years. In a previously-published companion paper, we first introduced an *hybrid haptic technique* which takes advantage of both the explicit and implicit geometric representation of objects [16]. This system provides a fast and stable haptic technique that avoids the force discontinuity without an exaggerate feeling of round object. In this paper, haptic decoration and material editing are implemented based on this haptic technique.

Recently, a few haptic systems [15, 8, 7] support haptic painting, where the user can paint directly onto a 3D model. When the user paints on the surface, corresponding portions in the texture map are updated. Using haptic painting, we can easily make correct and undistorted texture images on the desired area of the 3D model without knowledge about the mathematics of the surface parameterization. In this paper, we extend our previous haptic systems with a novel, image-based haptic texturing and editing technique for material properties like friction and stiffness. These ideas are integrated into our haptic decoration and material editing system.

Haptic decoration allows the user to paint directly on the surface (haptic painting) and then feels the paint thickness variation due to the painted image (image-based haptic texturing, see Figure 1). In haptic painting, a volume fill algorithm based on a volumetric representation is used to find the 3D triangles to be rasterized within the 3D brush volume. Our image-based haptic texturing is implemented by modulating potential values in the volumetric implicit surface representation resulting in a new geometry of the implicit surface. By simulating the textured implicit surface rather than a geometric model, the user correctly senses the textured surface regardless of the texture mapping method and the complexity of the geometric model.

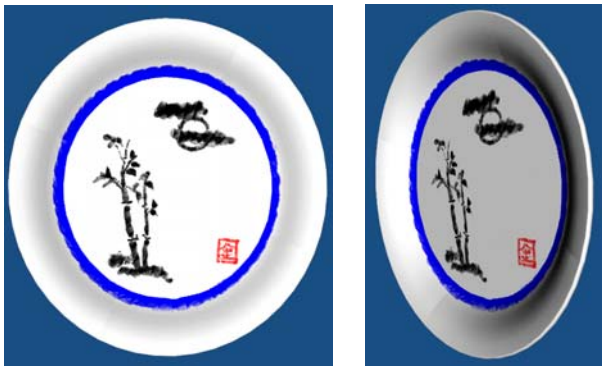


Figure 1: An Asian-style plate decorated by our haptic system

Material editing enables the editing and sensing of local surface properties like stiffness and friction on a 3D model instead of applying global properties to the model as in most previous approaches. While the user performs material editing, a material map containing values of local surface properties is saved into a volumetric representation rather than vertices of the geometric model. The system simulates the surface properties by interpolating values of local material properties in a cell containing the virtual contact point on the surface. As a result, material editing is performed regardless of the complexity of the geometry model.

Haptic painting and editing material properties are performed in the visual display loop (at 30Hz). Meanwhile sensing the textured surface and material properties are implemented in the fast haptic servo loop (at 1 KHz).

This paper also introduces some novel features in hap-

tic display. First, the system uses an offset surface to provide additional internal volume to prevent the tool tip from passing through thin objects without introducing any force discontinuity. Haptic decoration and material editing can also be performed on thin objects using the offset surface (see Figure 1, 9). Next, a magnetic surface is used to force the tool tip to stick to the surface while exploring the 3D model. Finally the system can simulate multiple objects by merging their implicit surfaces into a single implicit surface which represents combined multiple objects.

1.1 Advantages of implicit surface representation

The implicit representation of the surface of a 3D object is traditionally described by an implicit equation [5]. Our haptic rendering algorithm uses a *discrete, volumetric implicit surface representation*, where the implicit function defining the surface is only sampled on a regular 3D grid [16]. Such a surface representation has the following advantages in haptic rendering:

- Fast collision detection and contact point determination on the surface using the potential value in each grid point to indicate the proximity to the surface.
- Fast surface normal computation through the gradient of potential values.
- Easy offset surface computation as iso-surfaces of the same potential with different iso-values.
- Constant complexity of force computation, even for complex 3D surfaces since the force computation is performed only locally in the cell containing the tool tip.
- Reduced numerical issues: raw and/or complex geometry models often have small gaps at a common edge or vertex due to small numerical errors [24]; these may cause the tool tip to fall inside the internal volume suddenly through the gaps, which could not happen for an implicit-based technique.

1.2 Contributions

There are several contributions of our system. These are as follows:

- we first suggest a material editing technique using haptic display instead of a global material property on a 3D model.
- we propose a fast and stable image-based haptic texturing method, that allows the user to sense the surface thickness variation.
- the offset surface is used to give an additional internal volume to thin objects for generating constrain force. Therefore, the system avoid the force discontinuity unlike constraint-based approaches [30, 24].
- We introduce a magnetic surface in our haptic system which forces the tool tip to stick to the surface while exploring a 3D geometric model.
- The system simulates multiple objects by merging their implicit surface representations into a new implicit surface representation.

We will proceed with a short discussion on previous related work in Section 2. Our basic and advanced force models are described in Section 3 and methods of haptic decoration and material editing are given in Section 4 and 5 respectively. In Section 6, we describe implementation and results of our haptic editing techniques. Finally we conclude the paper with a discussion of future work in Section 7.

2 Related Work

2.1 Haptic rendering algorithm

Traditional haptic rendering methods can be classified into roughly three groups according to the surface representation used, be it geometric, volumetric (including implicit), or parametric.

One approach for the geometric models is the penalty-based method [18, 19] which suffers from a strong force discontinuity and passing through thin objects. In order to overcome these limitations, Zilles and Salisbury [30] introduced a constraint-based "god-object" method in which the movement of a god-object (a virtual contact point on the surface) is constrained to stay on the object's surface. Ruspini et al. [24] proposed a more general constraint-based method and simulating additional surface properties such as friction and haptic texture. The constraint-based approach, however, still suffers from force discontinuity and does not scale well with the complexity of the object's surface. Force Shading solves the force discontinuity problem, but introduces an amplified feeling of rounded surfaces due to the discrepancy between the haptic force field and the actual normal field of the surface [12, 24].

In the haptic rendering for volumetric data, the force field is computed directly from the volume data without conversion to geometric model. A haptic rendering algorithm for volumetric data was first introduced by Iwata and Noma [14] and was improved in [2, 17]. They used the gradient of the potential value to calculate the direction of the force and the amount of force was linearly proportional to the potential value. Salisbury and Tarr [26] suggested a haptic rendering algorithm for implicit surfaces defined by analytic functions. This algorithm is, however, not applicable to geometric models with arbitrary shapes.

Thomson et al. [28] introduced a haptic system to directly render *NURBS* model using the advantage of mathematical properties of parametric surface representation.

This system is implemented in multiple stages: surface proximity testing to find the distance from the tool tip point, local tracking of the motion of the tool tip along the parametric surface, and contact processing to generate the constraint force. Some haptic systems [27, 15] also were proposed to render *NURBS* models.

2.2 3D Painting

Hanrahan and Haeberli [10] first suggested a 3D painting system using a 2D mouse interface which allows the user to paint directly onto a 3D model. The Chameleon system [13] introduced an advanced painting system which dynamically generates a tailored UV-mapping for newly painted polygons during the painting process. The drawback of painting systems incorporating a mouse interface [10, 13] is that they can not provide a natural and precise painting style as well as force response. A 3D painting system with a 3d tracker sensor [1] has the issue that the user should look at both the real object and scanned model at the same time to see whether the paint is being applied correctly. This registration process may make the painting style awkward. In addition, the system requires real objects to perform the painting.

Johnson et al. [15] first proposed a haptic painting system which paints directly onto trimmed *NURBS* models with a haptic interface. Incorporating a haptic interface provides a natural painting style as if the user paints on a real object. While the user paints on a 3D model, the system updates the 2D texture map and adaptively adjust the brush size using the surface parameterization to mitigate the texture distortion between the 2D texture image and the 3D model. However, this system works only for *NURBS* models and performs a flood fill on the region within the brush volume in the texture space resulting in texture distortion on complex 3D models.

A haptic painting system (called *inTouch*) for polygonal models was developed by Gregory et al. [8]. In addition to painting, the system can edit polygonal models of arbitrary topology at multiple resolution, based on a subdivision surface representation. In order to avoid the texture distortion problem, they use a standard 2D scan-conversion in texture space. During the scan-conversion, each texel in the 2D triangle in the texture space is updated according to a brush function based on its corresponding 3D position. Building upon the framework of *inTouch*, *ArtNova* [7] allows the user to patch textures interactively to a local region by brush strokes and the orientation of the texture is determined directly by the stroke. Baxter et al [4] suggested a painting system on 2D virtual canvas using deformable 3D brushes.

2.3 Image-based haptic texturing

In addition to visual realism of the graphical texture, image-based haptic texturing enhances haptic realism at the same time. Indeed, image-based haptic texturing allows the user to feel the ”height” variations of a 2D image (thickness of the paint layer).

Ho et al. [11] implemented an image-based haptic texturing by perturbing the direction and magnitude of the surface normal using height fields. The height field of any point on the surface is generated from the texture map using a two-stage texture mapping method. In order to perturb the surface normal, they use a local gradient of the height value. This image-based haptic texturing, however, requires additional computations for detecting collision between the tool tip and a new textured surface and perturbing the surface normal. In addition, the direction and amount of force is computed based on the height value at the virtual contact point (we will call it “VCP”) [the white circle in Figure 6] on the original surface instead of the textured surface. This makes the system inaccurate because of the sudden changes in the magnitude and direction of force.

3 Implicit-based Haptic Rendering Technique

In this section we describe our implicit-based haptic model and some novel features such as the offset surface for thin objects, magnetic surface, and multiple objects simulation.

3.1 Haptic model

We first briefly summarize our implicit-based haptic model; the detailed algorithm was described in [16].

Collision detection. In the implicit representation, collision detection becomes trivial due to the inside/outside property of the implicit surface. If the potential value at the position of tool tip changes sign from positive to negative, a collision is detected (the iso-surface is, by convention, defined as zero).

Force generation. The force direction is computed by interpolating the gradients of 8 cell points around the tool tip. The interpolation function makes the system avoid the force discontinuity. In order to determine the amount of force, we first find the virtual contact point (VCP) on a surface which is the intersection point between the surface and a ray along the computed force direction. The amount of force is proportional to the distance between the VCP and the tool tip. Once the VCP is determined, a spring-damper model [29] is used to compute the final force vector that tries to keep the virtual contact point and the tool tip at the same position.

Adding friction to the model. Simulating friction is implemented by limiting the movement of the virtual contact point like in the constraint-based method. In addition to surface friction, the system takes into account the depth of penetration. It means that as the tool tip penetrates deeply inside the surface, the user feels higher friction.

Implicit-based haptic texture. In previously introduced algorithms, haptic textures are implemented by modulating the surface friction and/or local surface normals. In our algorithm, haptic texturing is simulated by applying texture values *directly to the potential value of each point in the 3D grid*, without any need for preprocessing. No modification to the existing algorithm is necessary in order to accommodate the new texture features of the surface. Moreover, there is no additional computational cost imposed due to haptic texturing since the direction and amount of the force are computed dynamically as the tool tip moves along the surface, whether or not this one has been modified by additional textures.

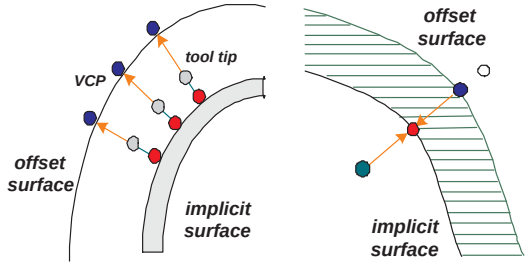
3.2 New offset surface for thin objects

In penalty methods, if 3D models have not sufficient internal volume, the haptic system can not generate the enough constraint force to prevent the tool tip from passing through the models. Similarly, this problem can be occur in our approach. Constraint-based approaches [30, 24] address this problem by limiting the movement of the VCP by the surface but introduce strong force discontinuity resulting in a feeling of bumpy surface around volume boundaries such as edges and vertices in geometric models.

Figure 2: Haptic simulation on an object (Galleon: 8794 triangles) with thin volume using an offset surface

In order to give thin objects sufficient internal volume without force discontinuity, we use an offset surface which represents an iso-surface with a positive offset from the implicit surface. An additional volume between the offset surface and the original surface allows the system to generate the appropriate constraint force for thin objects.

Note that the system uses the surface normal as the force direction and surface properties at the closest point on the original surface (red circles in Figure 3a) from the tool tip to simulate the original surface as accurately as possible. The closest point is also used to represent the visual contact point on the surface and to update texture map and material map when the user performs haptic editing (see section 4, 5) on the offset surface. The force magnitude is proportional to the distance between the physical tool tip and the VCP on the offset surface (grey and blue circles respectively in Figure 3a). The offset surface could smooth out small dent on the surface but the system provides reasonable haptic fidelity in high resolution of volumetric implicit surface representation.



(a) (b)

Figure 3: Using offset surface (a) to provide additional volume for thin objects, or (b) to simulate magnetic surface

3.3 Magnetic surface

In previous work, Virtual Fixtures [23] were used to assist the operator as guides in the virtual environment by restricting the motion of the end-effector. These help the operator to execute tasks quickly and precisely. Similarly, the force field in the Haptic Buck system [6] attracts the haptic device to the closest control handle to assist the user to perform a designed control in 3D virtual environment. In addition, this system forces the haptic device to follow the mechanism trajectory by constraining motion.

We propose a magnetic surface which attracts the tool tip to the closet point on the surface if the tool tip is within a magnetic field. Unlike previous two systems, the purpose of the magnetic surface is to force the tool tip to keep in contact with the surface while the user explores complex 3D models. It helps the user, especially visually impaired people, to feel the shape of 3D model without losing the contact with the surface.

The magnetic field is created in a narrow band between the offset surface and the original surface (hatched area in figure 3b). The direction of magnetic force is determined by the surface normal at the position of tool tip and the amount is proportional to the distance between the tool tip and the

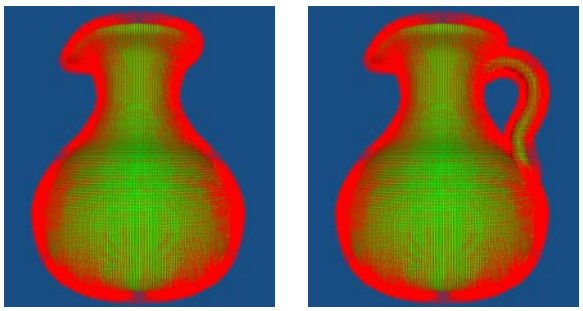
closest point on the original surface (see Figure 3b). The user can adjust the range and strength of the magnetic force according to applications.

3.4 Merging multiple objects

Complex objects are often constructed by simply adding or merging multiple objects together (see Figure 4). When the tool tip is near a region where several objects overlap, the VCP should still smoothly shifts onto the outmost surface of the new object. Dealing with this transition is not trivial and often not clear in previous algorithms.

In our algorithm, when multiple objects are combined, corresponding implicit representations are merged resulting in a new implicit surface representing multiple objects. At this point, the minimum one among potential values is chosen as the final potential value for the new implicit surface. It works like a union point-set operation in CSG.

The system use the new implicit surface representation to simulate multiple objects and does not need to take care of transition of the virtual contact point among multiple objects and modification of the algorithm (see Figure 4).



(a) (b)

Figure 4: Merging two implicit surface representations to simulate multiple geometric objects:(a) initial object (b) adding a handle to the object.

4 Haptic Decoration

In this section, we describe *haptic decoration*, that allows the user to paint directly on the surface, and then to sense the surface variation of the painted image on the surface.

4.1 Haptic Painting

We use a volume fill algorithm to find the 3D triangles to be painted within the brush volume. 2D triangles in texture space are rasterized in a similar way as in *intouch* [8] during haptic painting. A simple brush function is used to update each texel in a texture map during the rasterization. We explain these two points in more details next.

Finding 3D triangles to be rasterized. 2D triangles being painted in texture space are determined by corresponding 3D triangles which fall within the brush volume in 3D space. In our system, the volumetric representation is used to find 3D triangles to be painted. Each grid point has an index of the nearest face which is computed while building the implicit surface representation before haptic simulation. These face indices are used for the connection between the implicit surface and geometric model.

In order to find the seed face to start the rasterization, the system performs an intersection test between a line segment from the tool tip position to the VCP and faces in the grid cell containing the VCP (see Figure 9c). Then, the system walks through all faces within the brush volume from the seed face. If a face is within the brush volume and is not in the list of faces being painted, the face is added into the list. The process to find faces being painted is repeated recursively until no new face is found (this process is known as a *flood-fill algorithm*).

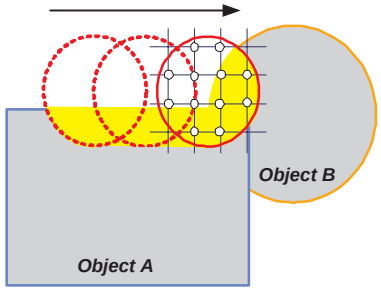


Figure 5: A volume-fill algorithm to find 3D triangles within the brush volume (red circle representing the brush volume, white points indicating the grid points within the brush volume, yellow area being painted).

However, this flood-fill algorithm can not find all triangles within the brush volume when the tool tip is moving on the overlapped area of multiple objects. For the volume-fill, the system checks all faces indicated by grid points (white points in Figure 5) within the brush volume. If it finds a new face within the brush volume, this face is used to a new seed face to perform another flood-fill on a new object (called *volume-fill algorithm*). As a result, we can make the list containing all faces within the brush volume.

Triangle rasterization and brush function. The system finds 2D triangles in texture map using the face list being painted. These 2D triangle are rasterized one by one using a standard scan conversion (see Figure 9b). The color of each texel are determined by a function of the brush size, brush color, and fall-off rate, background color and the distance to the center of the brush volume during the triangle rasterization. Previous 3D painting systems [10, 1, 8] have

suggested similar brush functions. Equation 1 is used to compute the weighting factor $\alpha \in [0, 1]$ for standard alpha blending between the current texel color and the brush color as follows:

$$\alpha = 1 - (\frac{R_t}{R_b})^f \tag{1}$$

where R_t is the distance to the center of the brush from corresponding 3D position of the texel, R_b is the radius of brush volume, and f is a fall-off rate which is specified by the user. The size of brush volume is determined according to the amount of applied force or a user-specified size. 3D position for each texel is computed from a 2D triangle containing the texel using a barycentric coordinate. The blended color of a texel is computed by Equation 2.

$$\begin{aligned} C_f[R,G,B] = C_b[R,G,B] * \alpha \\ + C_t[R,G,B] * (1-\alpha) \end{aligned} \tag{2}$$

where $C_b[R,G,B]$ is the brush color (foreground) and $C_t[R,G,B]$ is the current texel color (background).

4.2 Image-based haptic texturing

We use the implicit-based haptic texturing [16] to sense the surface variation of 2d image (called *image-based haptic texturing*). While the image-based haptic texturing applies, all potential values in grid points around the surface are updated according to corresponding texture values at a time. In order to obtain texture value of each grid point, the system first finds the closest 3D position on the closest face from each grid point. Then, the 3D position is mapped into a 2D texel in a texture map by barycentric coordinate. Finally, the texture value is obtained from the grayscale intensity of the texel. After updating potential values, we get a new geometry of implicit surface on which the user correctly senses the surface variation. This method is independent of the shape complexity and texture mapping method unlike in previous systems [11, 24]. The system also does not introduce the sudden change in the force direction and amount since the force is generated based on the textured geometry instead of the original geometry (see Figure 6).

Haptic painting is performed in the visual display loop. Meanwhile image-based haptic texturing is simulated in the fast haptic loop without delay which often causes to miss significant change in texture. Furthermore, no addition computational cost and change in the algorithm are required.

5 Material editing

Texture map in computer graphics is used to make the surface more realistic by locally changing the color or appearance of the surface. Similarly, the user may expect haptically different surface properties when he/she touches a graphically textured model since typically different material have different surface properties like stiffness and friction. However, in most previous haptic systems, a single surface property is applied over the whole 3D model.

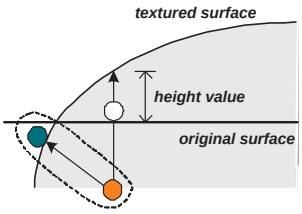
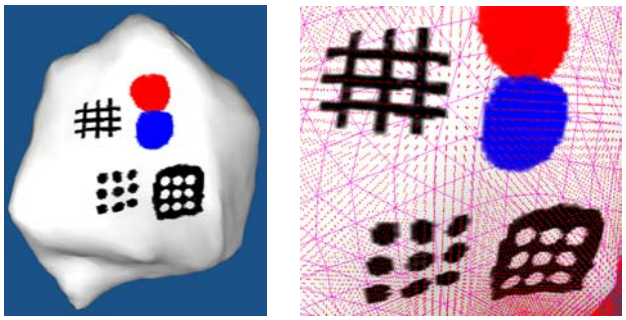


Figure 6: A previous image-based haptic texturing method and our approach (dotted line)

Recently, Pai et al. [22] suggested a contact texture including a friction map which is saved in each vertex in a 3d model. Friction values are measured by scanning the response of real objects with a robot system. However, it does not support editing the friction map.



(a) (b)

Figure 7: Various haptic effects using material editing (a) the user edits and feels the material properties while painting (b) a closer view showing the mesh and the volumetric representation

We first suggest a material editing technique to edit and simulate material map (see Figure 7). Material map contains local surface properties such as stiffness and friction coefficients just as texture map has texture values. In our method, the user performs haptic painting and material editing directly on the surface at the same time. During the editing of material properties, local material properties are saved at grid points within the brush volume in a 3D grid instead of vertices in geometric models (see Figure 7b).

After editing the material map, the user can sense the assigned local surface properties on the surface immediately. As the tool tip moves on the surface, the system

computes corresponding friction and stiffness values at the VCP by linearly interpolating material properties at 8 grid points in a cell containing the VCP on the surface. This interpolation function gives a fast and reasonable approximation. Based on the volumetric representation, material editing technique is independent of the geometric model. The material map can be extended to other haptic properties like surface roughness for instance.

6 Implementation and results

The haptic algorithms in this paper have been implemented on a PC with Pentium III and E&S AccelGALAXY 16mb video card. A 3DOF PHANTOM was used for haptic interface. The volumetric representations for all 3D models in this paper are sampled on a 150x150x150 grid. Haptic simulation constantly runs at about 1kHz based on the GHOST library to send the generated force directly to the PHANTOM device. As we mentioned before, the shape complexity of 3D models and the grid resolution for the volumetric representation do not affect haptic performance, given that the grid all fits in memory.

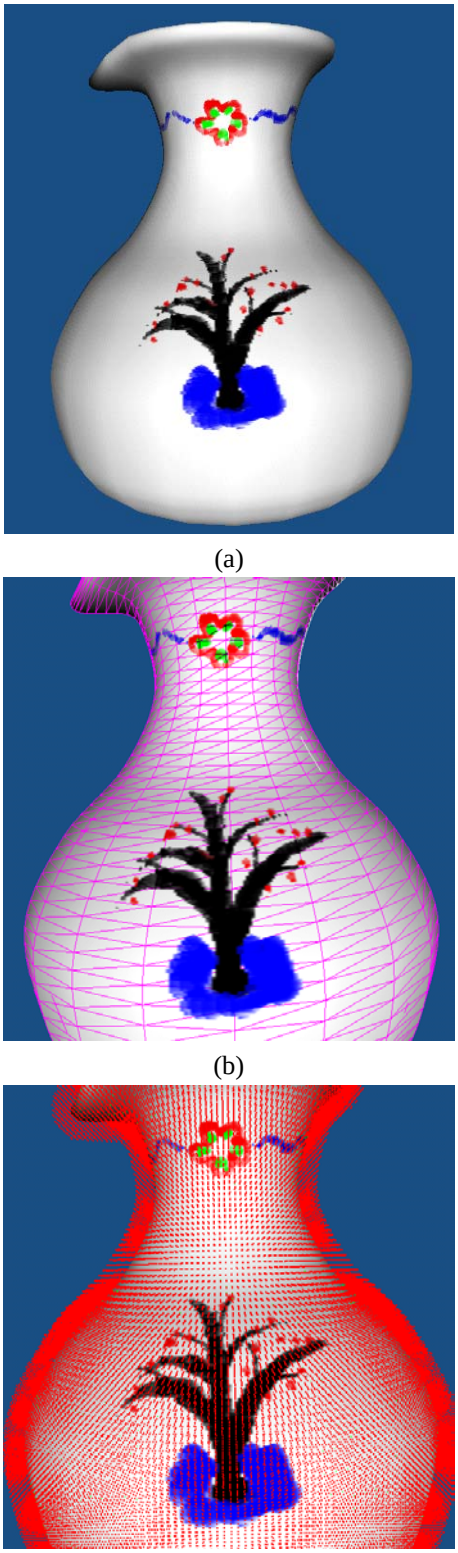
Haptic decoration and material editing techniques allow the user to paint and edit material properties directly on the 3D model in a natural way. In addition, the user can feel the surface variations due to painted image and material properties assigned. Figure 8, 9 show some examples edited by our system. Especially, a pottery model in Figure 9 is edited based on the offset surface. For the haptic painting, we use a parameterized texture map with 512x512 pixels.

(a) (b)

Figure 8: Haptic editing examples (a) a panda (b) a self-portrait

7 Conclusion and future works

Our haptic decoration technique allows the user to paint directly on the surface and then to sense the surface variation of the painted image. In a similar fashion, the user can edit material properties such as friction and stiffness over a



(a)

(b)

(c)

Figure 9: Haptic decoration on the offset surface (a) a decorated pottery (4800 triangles) (c) the pottery model with its wire-frame (d) the pottery model with its implicit surface representation

3D model instead of applying global properties over a whole model. Then, the user can feel the surface properties in the volumetric representation on the surface.

In addition, we have extended the initial haptic technique, for example offset surface to provide additional internal volume for thin object without introducing force discontinuity, magnetic surface to force the tool tip to stick to the surface while exploring complex 3D models, and merging multiple implicit representation to simulate multiple objects just as the system simulates a single object.

In the future, we plan to develop a 3D embossing and engraving system using our image-based haptic texturing method. We also plan to reduce the memory requirement using an octree data structure.

Acknowledgments The work reported here was supported in part by IMSC NSF Engineering Research Center (EEC-9529152), by a NSF CAREER award (CCR-0133983), and two NSF CARGO awards (DMS-0221666, DMS-0221669).

References

[1] M. Agrawala, A. C. Beers, M. Levoy, "3D Painting on Scanned Surfaces", In symposium on interactive 3D graphics, 1995

[2] R. S. Avila, L. M. Sobierajski, "A Haptic Interaction Method for Volume Visualization" IEEE Visualization'96, Oct. 1996.

[3] C. Basdogan, C. Ho, M. A. Sriniviasan, "A Ray-based Haptic Rendering Technique for Displaying Shape and Texture of 3D Objects in Virtual Environments", Proceedings of the ASME Dynamic Systems and Control Division. 1997

[4] B. Baxter, V. Scheib, M. C. Lin, D, Manocha, "DAB:Interative Haptic Painting with 3D Virtual Brushes", In ACM SIGGRAPH 2001

[5] J. Bloomenthal, et al. "Introduction to Implicit surface", Morgan Kaufmann Publishers, Inc. 1997

[6] P. Buttolo, P. Stewart, A. Marsan, "A Haptic Hybrid Controller for Virtual Prototyping of Vehicle Mechanisms", ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 2002.

[7] M. Foskey, M. A. Otaduy, M. C. Lin, "ArtNova: Touch-Enabled 3D Model Design", IIn IEEE Virtual Reality, 2001

[8] A. Gregory, S. Ehmann, and M. C. Lin. "inTouch: Interactive multiresolution modeling and 3D painting with a haptic interface". Proceedings of IEEE VR Conference, 2000.

[9] A. Gregory, M. Lin, et al. "H-Collide: A Framework for Fast and Accurate Collision Detection for Haptic Interaction", In IEEE Virtual Reality, 1999

[10] P. Hanrahan, P. Haeberli, "Direct WYSIWYG Painting and Texturing on 3D Shapes", In ACM SIGGRAPH, pp. 215-223, 1990

[11] C. Ho, C Basdogan, M. A. Sriniviasan, "Efficient Point-Based Rendering Techniques for Haptic Display of Virtual Objects", Presence. Vol 8. No 5, October 1999, 447-491.

[12] J. M. Hollerbach and D. E. Johnson, Virtual Environment rendering, in Human and Machine Haptics, M. Cutkosky, R. Howe, K. Salisbury, and M. Srinivasan, eds., MIT Press, 2000.

[13] T. Igarashi, D. Cosgrove, "Adaptive Unwrapping for Interactive Texture Painting", in ACM Symposium on Interactive 3D Graphics, 2001.

[14] H. Iwata and H. Noma, "Volume haptization" In Virtual Reality, pp. 16-13. IEEE ' 93 Symposium on Research Frontiers.

[15] D. Johnson, T. V. Thompson II, M. Kaplan, D. Nelson, E. Cohen, "Painting Textures with a Haptic Interface", In IEEE Virtual Reality(VR), 1999

[16] L. Kim, A. Kyrikou, G. S. Sukhatme and M. Desbrun, "An Implicit-based Haptic Rendering Technique", IEEE IROS 2002, Switzerland.

[17] D. A. Lawrence, C. D. Lee, L. Y. Pao, "Shock and Votex visualization Using a Combined Visual/Haptic Interface", IEEE Visualization'00 2000

[18] W. Mark, S. Randolph, M. Finch, J. V. Verth, R. M. Taylor II. "Adding Force Feedbackto Graphics Systems: Issues and Solutions" In SIGGRAPH'96, pp. 447-452, August 1996

[19] T. H. Massie, J. K. Salisbury, "The Phantom Haptic Interface: A Device for Probing Virtual Objects" ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 1994.

[20] S. Mauch, "A Fast Alogorithm for Computing the Closest Point and Distance Transform" http://www.cco.caltech.edu/ sean/closestpoint/ closept.html.

[21] M. D. R. Minsky, "Computational Haptics: The Sandpaper System for Synthesizing Texture for a Force-Feedback Display", PhD thesis, MIT, June 1995

[22] D. K. Pai, K. Doel, D. L. James, J. Lang, J. E. Lloyd, J. L . Richmond, S. H. Yau, "Scanning Physical Interaction Behavior of 3D Objects", In SIGGRAPH'01, 2001

[23] S. Payandeh, Z. Stanisic, "On Application of Virtual Fixtures as an Aid for Telemanipulation and Training", ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 2002.

[24] D. C. Ruspini, K. Kolarov, and O. Khatib. "The haptic display of complex graphical environment" In ACM SIGGRAPH pp. 295-301, 1997

[25] K. Salisbury, et al., "Haptic rendering: programming touch interaction with virtual objects" Symposium on Interactive 3D Graphics, pp. 123-130, 1995

[26] K. Salisbury, C. Tarr, "Haptic Rendering of Surfaces Defined By Implicit Functions", ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems pp. 61-68, 1997.

[27] P. Stewart, Y. Chen, P. Buttolo, "Direct integration of haptic user interface in CAD systems", Proceedings of the ASME Dynamic Systems and Control Division. 1997

[28] T. V. Thompson II, and E. Cohen, "Direct Haptic Rendering of Complex Trimmed NURBS Models," ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems, 1999.

[29] A. Yoshitaka, T. Kumano, K. Ogino, "Intermediate Representation for Stiff Virtual Objects", IEEE VRAIS'95 pp. 203-210.

[30] C. Zilles, J.K. Salisbury, "A Constraint-based God-object Method For Haptic Display", ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 1994.