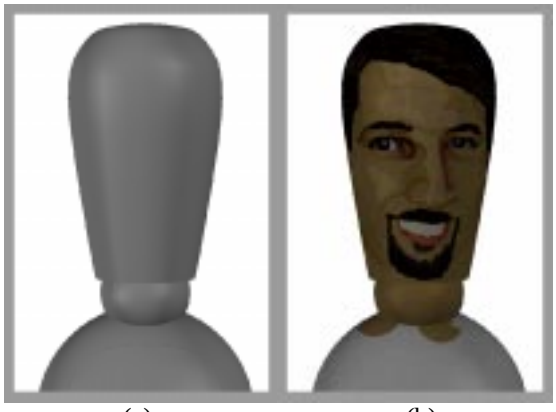
Painting Textures with a Haptic Interface

David Johnson, Thomas V Thompson II, Matthew Kaplan, Donald Nelson, Elaine Cohen
Department of Computer Science, University of Utah

Abstract

We present a method for painting texture maps directly onto trimmed NURBS models using a haptic interface. The haptic interface enables an artist to use a natural painting style while creating a texture. It avoids the traditional difficulty of mapping between the 2D texture space and the 3D model space by using parametric information available from our haptic tracing algorithm. The system maps user movement in 3D to movement in the 2D texture space and adaptively resizes the paintbrush in texture space to create a uniform stroke on the model.

1 Introduction



(a) (b)

Figure 1(a). Simple geometry. (b). Simple geometry with textures.

In this paper we address the creation of hand-drawn texture images on three-dimensional models. The value of textures derives from their ability to add visual complexity to a model without requiring increased geometric complexity (Figure 1) [1]. Textures also allow artists to paint effects that would be difficult to achieve by combining geometry, surface properties, and lighting alone.

Our texture painting technique applies to models constructed from NURBS -- piecewise polynomial tensor product surfaces. These sculpted models are particularly useful in the areas of design and animation, where textures have frequent application.

Hand-drawn textures can be difficult to create because textures undergo a mapping from 2D texture space to the 3D model space during the rendering process. This mapping depends on the underlying mathematical representation of the model and can result in a non-intuitive positioning of the texture on the model.

(a)

(b)

Figure 2: (a) Textures may undergo distortion when mapped onto a model. (b) Our system maintains a uniform model space stroke by adaptively resizing in texture space.

An additional complication arises from the distortion the image may undergo during the mapping process. In Figure 2a, the checkerboard texture map stretches at the top of the goblet and compresses at the stem. A single line in the texture highlights this effect.

Drawing by hand the distorted texture image that maps to a uniform line on the model would be very difficult (Figure 2b).

We propose to avoid these problems by painting directly on the surface with a haptic interface. Our system maps the location of the paint on the surface back into the texture image. The user does not need to know the relationship between the portion of model being painted and the underlying texture map or maps. In addition, the system adaptively sizes the brush in the texture image to maintain a uniform brush size on the model. This avoids problems with distortion between the texture image and the model.

An additional advantage of the system comes from the natural interaction style the haptic interface enables. Using the sense of contact provided by the force-feedback device, the user can draw on portions of the model that are not directly visible. The contact cues also aid in spatial positioning of the brush. The overall feel is that of painting a real object, but aided by the power of a computer.

2 Background

We follow the definition of texture mapping from [1], a mapping of a function onto a surface in 3D. In the case of a hand-drawn texture representing or modulating the model color, the function has a 2D domain, evaluates to an (r,g,b) point, and is stored in an array. Each point in the array is known as a *texel*. Texture mapping to a surface color was introduced by [2]; other mappings include normal vector pertubation [3], specularity [4], and transparency [5].

There have been two main approaches to the creation and application of hand-drawn textures. The first approach attempts to map textures in model space back into texture space to hide the effects of distortion. The second approach uses information about the surface to minimize the effects of distortion and placement from the mapping process [6,7,8]. Our approach falls in the first category.

2.1 3D paint

Methods that allow a user to paint onto the model rather than in the texture image space are commonly known as 3D paint programs. In [9], Hanrahan painted onto 3D mesh objects using a mouse. An auxiliary item buffer that stores an object ID for each pixel in the display provided an efficient means of determining what portion of the model lay underneath the mouse cursor. Paint and surface property information could then be stored with the mesh vertices.

This approach was extended to 3D input devices in [10]. Their system was intended for scanned physical objects --- paint colors are stored in the vertices of the scanned model while the registered physical model provides an object to paint against with a virtual brush. A Polhemus tracker determines the position of the paintbrush in space.

For both of these systems, the painting style can be awkward. In the first paper, the mouse can only paint on directly visible portions of the model. Orienting a model so that desired portions are visible can be a non-trivial task. In the second paper, the use of a physical object plus the need for registration between the mesh and object complicates the painting process.

Furthermore, both papers store the painted color information at the mesh vertices. This implies that added paint detail requires an increase in mesh complexity. Normally, a benefit of texture mapping is that it allows increased visual detail without added geometric detail; we would like to maintain that advantage.

3 System overview

Our 3D paint system is built upon a system for virtual prototyping of CAD models [11]. A basic functionality provided by the virtual prototyping system is tracing along NURBS models using a haptic interface [12]. We can use this functionality as a basis for directly painting textures onto a model.

3.1 Hardware

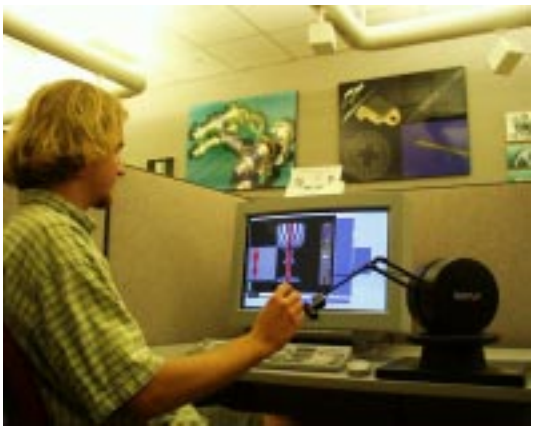


Figure 3: The paint system in use. A haptic interface allows 3D positioning and returns contact sensations.

The paint system (Figure 3) consists of three main components: an SGI Onyx2 with hardware texturing, an SGI Indigo2, and a haptic device. We currently allow either a high-resolution Phantom [13] or a high degree-of-freedom SARCOS Dextrous Arm as the haptic device [14].

3.2 Software

The SGI Octane workstation runs customized multi-threaded modeling and viewing programs [15] to display the painting environment. The Indigo2 workstation controls the Phantom and runs the haptic tracing code. This distributed model allows both the graphics and haptic tracing code to run with appropriately high update rates.

As the Phantom moves, the Indigo2 sends the Phantom's position to the viewing program. The viewing program determines rough proximity to models in the scene and returns that information to the tracing code on the Indigo2. The haptic tracing process on the Indigo2 tracks the closest point on the model at haptic rates using the direct parametric tracing (DPT) method [12]. Penetration into the model and the appropriate restoring force are determined using that tracked closest point. The haptic tracing process then sends the tracked closest point back to the viewer for use in the paint process.

4 Interface

The main portions of the paint interface are the toolbar, the paint ball, the model painting space, and the optional texture display square (Figure 4). All of the painting functions are accessed through the haptic interface.

Figure 4: The user interface: (left to right) the texture display, the model painting space, the paint ball, and the paint toolbar.

The haptic interface controls the 3D position of the paint ball. If the painter scrapes the paint ball across a color block on the toolbar, that color is slowly added into the current color, allowing the painter to "pick up" paint. Mixed colors can be saved into a few blank boxes at the bottom of the tool bar and colors on the model can be grabbed for reuse. Brush resizing is accomplished by touching the brush area on the tool bar and stretching or shrinking the paint ball radius.

Paint is added to the model by pushing the paint ball against the model. Currently, we use a simple color replacement scheme while adding paint --- more complex mixing methods would be appropriate as well.

5 Painting

Because we track the closest point on the model in parametric space we can easily determine the related texel in texture space to fill with color. The texture space is just a uniform discretization of the parameter space, so a linear mapping suffices to determine the correct texel.

5.1 Adaptive brush sizing

We would like to use paintbrushes of an arbitrary size in the 3D paint program. We cannot use constant distance measures in the texture space to set the brush size, or the brush will distort in the model space.

In order to fill a uniform brush in model space, we need to map potential texels onto the model and then measure their Euclidean distance from the Euclidean center of the brush. The mapping from texture space to model parameter space is a linear mapping; we then find the point on the model by evaluating the surface at that parametric point.

Surface evaluation on a NURBS surface can be a relatively expensive operation. Fortunately, we have invested time improving the performance of surface evaluation for previous haptic rendering work [11,12]. However, the number of surface evaluations still proved to be a bottleneck in our initial simple approach of testing every texel in a large block for Euclidean distance to the brush center.

Instead, we have developed an approach for finding the perimeter of the brush in texture space and then filling in all the contained texels with the brush color. We trace the perimeter by moving out from the center of the brush until the desired radius is reached, then searching neighboring texels for the brush boundary. The interior region is filled with color using a flood-fill method. With this approach, the number of surface evaluations grows linearly with the brush radius, better than the quadratic growth of block testing. We were able to maintain interactive painting rates even while painting on large (1024 x 1024) texture maps and with brush sizes containing several hundred texels.

6 Future work

Clearly, additional paint tools can be adapted from 2D paint systems to be used in the haptic paint

environment. Some of these tools may require modifications to work in the 3D environment. We have also started to explore different techniques for efficiently filling the brush area in the texture map with the goal of further increasing the performance of the system.

7 Conclusion

Figure 1 demonstrates the effectiveness of using the haptic 3D paint system. The textures on the model were painted entirely using our system.

The main feature of the paint system is its simplicity. It supports a natural style of painting texture onto a model by mimicking the normal painting process. By supporting direct painting onto the model and by adaptively sizing the brush the user does not have to be aware of the underlying surface patches, of the underlying surface parameterization, or of the mappings between textures and the model.

8 Acknowledgments

The authors would like to thank Bruce Gooch for video editing and the Virtual Prototyping Group for support. Lisa Durbeck generously let us use the Phantom to make the example textures in this paper. This work was supported by NSF grant MIP-9420352, DARPA grant F33615-96-C-5621 and by the NSF and DARPA STCCGSV grant (ASC-89-20219).

9 References

1 Heckbert, Paul, "Survey of Texture Mapping", IEEE CG and Applications. Nov. 1986, pp. 56-57.

2 Catmull, Ed, *A Subdivision Algorithm for Computer Display of Curved Surfaces*, PhD thesis, Dept. of CS, University of Utah, Dec. 1974.

3 Blinn, James. "Simulation of Wrinkled Surfaces", *Computer Graphics, (SIGGRAPH '78 Proc.)*, Vol. 12, No. 3, Aug. 1978, pp. 286-292.

4 Blinn, James. *Computer Display of Curved Surfaces*, Dept. of CS, Univ. of Utah, 1978.

5 Gardner, Geoffrey, "Visual Simulation of Clouds", *Computer Graphics, (SIGGRAPH '85 Proc.)*, Vol. 19, No.3, July 1985, pp. 297-306.

6 Bennis, Chakib et al, "Piecewise Surface Flattening for Non-distorted Texture Mapping", *Computer Graphics, (SIGGRAPH '91 Proceedings)*, Vol. 25, July 1991, pp. 237-246.

7 Maillot, Jèrôme et al, "Interactive Texture Mapping", *Computer Graphics, (SIGGRAPH '93 Proceedings)*, Vol. 27, Aug. 1993, pp. 27-34.

8 Litwinowicz, Peter and Miller, Gavin, "Efficient Techniques for Interactive Texture Placement", Computer Graphics, (SIGGRAPH'94 Proceedings), Vol. 24, No. 4, July 1994, pp.119-122.

9 Hanrahan, Pat and Haeberli, Paul, "Direct WYSIWYG Painting and Texturing on 3D Shapes", Computer Graphics, (SIGGRAPH'90 Proceedings), Vol. 24, No. 4, August 1990, pp.215-223.

10 Agrawala, Maneesh et al, "3D Painting on Scanned Surfaces", 1995 Symposium on Interactive 3D Graphics, Monterey, CA, pp.145-150.

11 Hollerbach, J.M et al, "Haptic interfacing for virtual prototyping of mechanical CAD designs," ASME Design for Manufacturing Symposium, (Sacramento, CA), Sept. 14-17, 1997.

12 Thompson II, T.V. et al, "Direct Haptic Rendering of Sculptured Models," Proc. Symp. on Int. 3D Graphics, pp. 167-176, April 27-30, 1997.

13 Massie, Thomas H., "Design of a Three Degree of Freedom Force-Reflecting Haptic Interface.", SB thesis, MIT EECS Department. May, 1993.

14 Jacobsen, S.C. et al, "High performance, high dexterity, force reflective teleoperator," *Proc. 98t^h Conf. Remote Systems Technology*, Washington, D.C., Nov. 1990, pp. 180-185.

15 Riesenfeld, R. et al, "Using the Oslo Algorithm as a Basis for CAD/CAM Geometric Modelling," *Proc. Nat. Computer Graphics Association*, 1991.