A Touch-Enabled System for Multiresolution Modeling and 3D Painting*

Stephen A. Ehmann **Arthur D. Gregory** **Ming C. Lin**†
Department of Computer Science
University of North Carolina
Chapel Hill, NC 27599-3175
{ehmann,gregory,lin}@cs.unc.edu
http://www.cs.unc.edu/∼geom/inTouch/

Abstract

We present an intuitive system, *inTouch*, for interactively editing and painting a polygonal mesh using a force feedback device. An artist or a designer can use this system to create and refine a three-dimensional multiresolution polygonal mesh. The appearance can be further enhanced by directly painting onto its surface. The system allows users to naturally create complex forms and patterns aided not only by visual feedback but also by their sense of touch.

Keywords: Haptic rendering, multiresolution modeling, 3D painting, shape deformation, H-Collide, virtual reality interface.

*This research is partially supported by ARO DAAG55-98-1-0322, NSF EIA-9806027, NSF DMI-9900157, NSF IIS-9821067 and DOE ASCI Award. The preliminary version of this work appeared in the Proceedings of IEEE Virtual Reality Conference 2000.

†Corresponding author, Office: 919-962-1974, Fax: 919-962-1799.

Figure 1: A Rooster Created & Painted by *inTouch*

1 Introduction

Simple, easy and fast model creation remains a challenging problem for computer animation and simulated environments. An ideal modeling package should allow the users to create a basic model, edit it with finer details, and further enhance its appearance by painting colors and textures with relative ease and flexibility – all via an intuitive and simple user interface. There are numerous commercial modeling systems available for computer animation, CAD/CAM, and virtual reality (VR) applications. It is also possible to scan in a model of an existing object or a prototype sculptured by an artist using (semi-) automatic digitizing systems.

One of the limitations of existing commercial modeling systems is lack of direct model interaction by using typical 2D input and output devices offered by current desktop computing environments. The resulting high learning curves deter many artists from freely expressing their creativity, due to the difficulty in translating conceptual designs into digital form.

In the computer animation, visualization, VR and user interface communities, researchers have developed numerous techniques for 3D interaction comprising object selection [PFC+97], flying, grabbing and manipulating [RH92], miniature worlds [PBBW95], different modes of speech, gesture and gaze, two-handed interaction [CFH97, ABF+97], and proprioception [MBS97]. Most of this work has focused on interaction techniques and is based on data-gloves or simple VR interfaces for selection and movement rather than on force-feedback devices. In the last few years, 3D input and output devices have been gaining importance. Haptic devices have been used as a virtual reality interface to simulated environments. The capability of "force feedback", introduced in systems like the Nanomanipulator [TRC+93] and the PIT [APT+98], offers direct interaction with virtual environments via the sense of touch.

As an attempt to provide touch-enabled modeling features, a non-commercial plug-in to Alias|Wavefront's Power Animator software package was developed at SensAble Technologies [Mas98]. Though its ability to feel the objects while placing or adding new objects provides a useful guide in model generation, "topological constraints of the NURBS geometry" prevented this approach from achieving desired force update rates [Mas98]. Not until very recently, a modeling system called $FreeForm^{TM}$ was introduced by SensAble Technologies. It is probably the first 3D digital modeling tool that al-

lows artists and designers to express their creativity with *3D-Touch*™. This system brings together the "sculpting" metaphors and the electronic modeling flexibility that make the sculpting of "digital clay" possible. The software package provides numerous features and works in a direct and obvious manner. However, the *FreeForm*™ modeling system uses an internal volumetric representation and associated techniques, so it appears rather difficult and time-consuming to create fine details and sharp features on models using this system. Furthermore, the system requires a representation conversion from its volumetric data structure to a desired surface representation.

Independently, we have been pursuing a similar vision of digital modeling with a true 3D interface via force-feedback devices. However, our approach is rather different. We have chosen subdivision surfaces as the underlying geometric representation for our system. This representation enables the user to perform global shape design and multiresolution editing with ease, allows the users to trade off fidelity for speed, and operates on simple triangular meshes. In addition, our system also offers 3D painting capability on arbitrary polygonal meshes with the haptic stylus as an "electronic paintbrush".

Our system, *inTouch*, can be used as a geometric modeler, where the user may load a simple primitive such as a triangle mesh approximation of a sphere and deform it to create an interesting model. Or, it can be used as a finishing system in conjunction with an existing modeling package by creating sharp features and finer details, as well as painting color and textures directly onto the model's surface. The rooster in Figure 1 was created and painted by *inTouch* and the sharp peak was effortlessly modeled by a simple pulling operation on the finest level mesh. Alternatively, *inTouch* can also be used as a natural and intuitive editing and painting tool to modify and refine a model scanned by a model digitizer. It complements existing techniques and modeling software.

1.1 Our Contributions

In this paper, we present an integrated system for 3D model editing and painting with a haptic interface. The system has the following characteristics:

- **Direct 3D model interaction with a haptic interface** – We use a commercial force-feedback device (PHANToM), a haptic toolkit (GHOST) and our collision detection library, H-Collide [GLGT99], to interact with a virtual model by directly manipulating points on the model surface and placing paint on the desired location with relatively high fidelity (limited by pixel precision).
- **Multiresolution model editing** – Based on a subdivision surface representation, we can shape and edit models of arbitrary topology at varying levels of detail. Due to the uniformity of representation, the resulting meshes can be used directly for rendering and simulation without any format conversion [SZ98].
- **Interactive 3D painting** – Given true 3D interaction, we can now paint directly onto the surface of the model without cumbersome mapping schemes or other object registration problems.

1.2 Organization

The rest of the paper is organized in the following manner. Section 2 presents related work in haptic interfaces, geometric modeling, and 3D painting. In Section 3, we describe an overview of the hardware configuration and software system framework used in this research. Section 4 and Section 5 present the design and implementation of the multiresolution modeling and 3D painting subsystems of *inTouch* respectively. We demonstrate the system capability and summarize user feedback in Section 6. Finally, we conclude with future research directions.

2 Previous Work

In this section, we briefly survey related research on haptic interfaces, geometric modeling, and 3D painting. Our system combines together interaction techniques, algorithmic advances in modeling, and 3D painting and integrates them seamlessly to develop an easy-to-use modeling and painting tool.

2.1 Haptic Interaction

Several real-time virtual environment systems have incorporated a haptic interface to enhance the user's ability to perform interaction tasks [CB94, Col94, FFC+95, MRF+96, MS94, OY90, RKK97, TRC+93]. Researchers at the University of Utah have developed a haptic display system for virtual prototyping [NNHJ98, He97, JC98]. It uses the Sarcos Dexterous Arm Master (with a 3-dof gripper) along with the *Alpha_1* CAD system, dynamic simulation, and haptic interface control.

Gibson [Gib95] and Avila and Sobierajski [AS96] have proposed algorithms for object manipulation including haptic interaction with volumetric objects and physically-realistic modeling of object interactions. Recently, SensAble Technologies developed the $FreeForm^{\mathsf{TM}}$ modeling system to create and explore 3D forms using volumetric representations [ST99].

2.2 Geometric Modeling

There is an abundant wealth of literature on geometric modeling, interactive model editing, and deformation methods applied to free-form curves and surfaces. They can be classified as pure-geometric representations [Far90] such as NURBS [PT97], free-form deformation (FFD) [SP86], or physically-based modeling techniques such as D-NURBS[QT96].

FFD [CR94, Coq90, HHK92, MJ96, SP86] is versatile and powerful for global shape design, but less efficient for local surface design. Hierarchical editing based on classical multiresolution analysis was first proposed to describe hierarchical geometry by Forsey and Bartels [FB88]. With a similar mathematical framework, subdivision methods allow modeling of arbitrary topology surfaces [SZ98], while supporting multiresolution editing [DKT98, HDD+94, KS99, SZMS98, ZSS97]. There are also other sculpting techniques based on volumetric modeling methods [GH91, RE99].

2.3 3D Painting

By using standard graphics hardware to map the brush from screen space to texture space, Hanrahan et al. allow the user to paint directly onto the model instead of into texture space [HH90]. This approach was applied to scanned surfaces using 3D input devices, such as data gloves and a Polhemus tracker [ABL95]. However, the painting style of both systems can be awkward, due either to the difficulty in rotating an object for proper viewing during painting, or to the deviation in paint location introduced by the registration process.

There are also commercial 3D painting systems. Most of them often use awkward and non-intuitive mechanisms for mapping 2D textures onto 3D objects. None offers the natural painting style desired by artists and designers.

Johnson et al. introduced a method for painting a texture map directly onto a trimmed NURBS model using a haptic interface [JTK+99]. Its simplicity and intuitive interface support a natural painting style. However, its parameterization technique is limited to NURBS and does not apply to polygonal meshes, which are more commonly encountered in computer graphics.

3 System Overview

Next, we give a brief description of our haptic system setup as well as the software libraries used in building the *inTouch* system and its user interface.

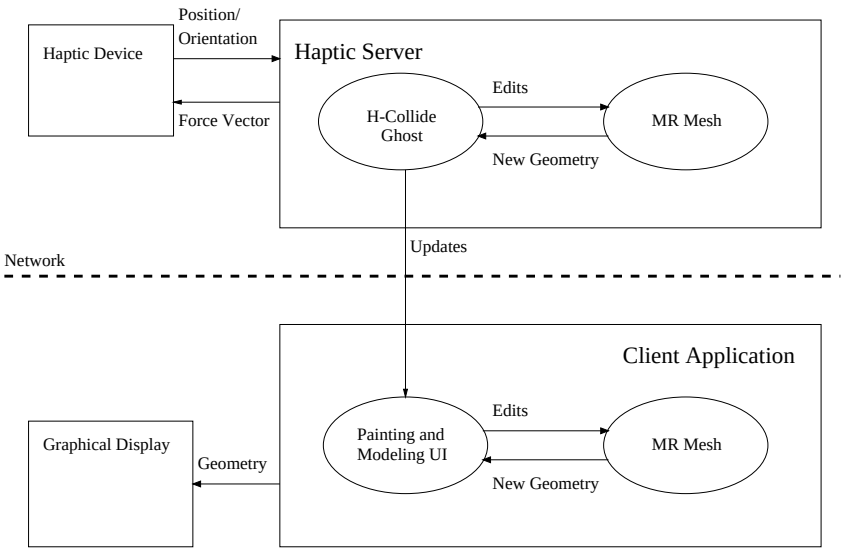

Figure 2: System Architecture

3.1 Haptic System Architecture

Our prototype system uses a SensAble Technologies' PHANToM as a haptic device, an Silicon Graphics Inc. R12000 Infinite Reality for graphical display, a dual-processor Pentium III PC as a haptic server, and UNC's VRPN library [VRPN] for a network-transparent interface between application programs and our haptic system. The system is written in C++ using the OpenGL and GLUT libraries.

The haptic server is composed of two basic processes. One process is used entirely by *GHOST* and H-Collide to update the force displayed by the PHANToM. The other handles the message passing across the network to the client application and also handles the model deformations. The client application is responsible for the graphical display and the user interface. The overall system architecture is shown in Figure 2.

The real-time haptic display is rendered using a commercial haptic toolkit called *GHOST* and our collision detection library, H-Collide [GLGT99, GLGT00]. All the collision queries are performed using H-Collide, which provides real-time contact determination for force computation at the KHz rate required for haptic display. Given a model, H-Collide pre-computes a hybrid hierarchical representation, consisting of uniform grids represented using a hash table and trees of tight-fitting oriented bounding box trees (OBBTrees). At run time, these hybrid hierarchical representations are used to exploit frame-to-frame coherence for fast proximity queries. If the model is deformed, all the hybrid hierarchical representations in the region of deformation are recomputed and updated in real-time. The contact information computed by H-Collide is also used for model editing and painting.

3.2 Software System

In order to deform the model interactively, the user simply chooses the edit level (resolution) and attaches the probe to the surface. The deformation update process uses the force vector currently being displayed by the PHANToM to move the current surface point at the selected edit level. These geometric changes are then propagated up according to subdivision rules to the highest level of the mesh. The changes are sent across the network to the client application which maintains

an identical multiresolution data structure so that it can perform the same operation to update the graphical display. We reduce the network traffic by performing the redundant computation of mesh deformation on both the server and the client machines, as opposed to sending all the changes to the highest level mesh across the network.

Once the highest level mesh has been modified, the H-Collide and graphical data structures need to be updated to reflect the change. The mesh data structure is queried for the triangles that were changed at the highest resolution, which is displayed both haptically and graphically at all times. A local deformation algorithm is used to merge the changed triangles with the triangles that were not changed in the H-Collide data structure. The graphical output subsystem also receives the update and proceeds to modify the display lists corresponding to the changed triangles and redraw the screen.

3.3 User Interface

inTouch allows the user to edit and paint a polygonal mesh with 3D haptic display. The projected 3D scene is composed of the model being edited and a model of the PHANToM probe. A 2D menu is drawn over the edge of this scene. The user can effectively interact with the 3D scene and the 2D menu without ever having to let go of the stylus.

In addition to file I/O, the user can position, orient, and scale the models with various 3D techniques. For shape deformation, the interface allows the user to pick the desired edit level and type of probe constraint. Probe constraints will be covered in section 4. Actual deformations occur by pressing and releasing the button when in contact with the model surface. The most recent mesh edit or brush stroke can be undone by simply double clicking the stylus button when not in contact with the surface.

For 3D painting, the user can interactively choose the color, saturation, and luminance of the brush stroke as well as its radius and falloff by naturally dragging in a 2D canvas. The haptic stylus returns the 3D location of the virtual editing tool and paint brush controlled by the user. It is drawn in the 3D scene as a sphere of radius equal to the effective radius of the virtual brush and colored the same color as the paint being applied. The radius is increased relative to the force exerted by the user on the haptic stylus as in real painting. Therefore, the harder the user presses on the virtual brush, the wider the paint spreads across the surface. A snapshot of the system setup with the main screen of the user interface is shown in Figure 3.

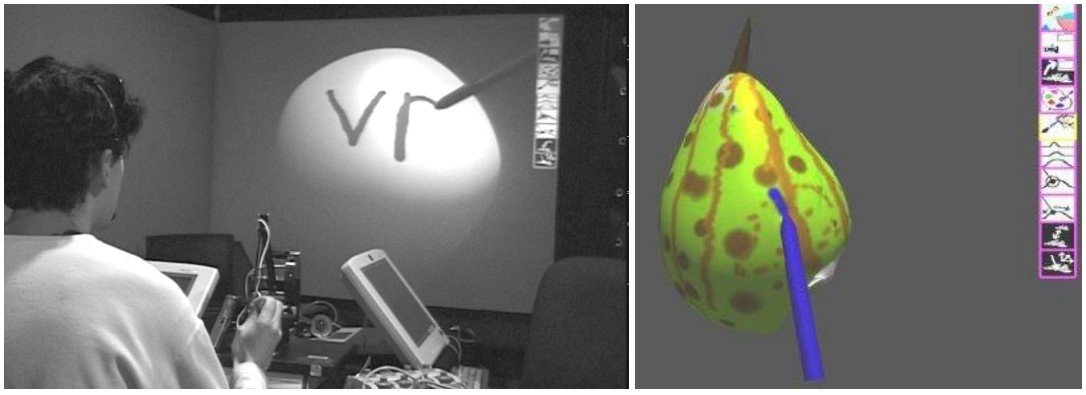

Figure 3: (a) Left: System Setup (b) Right: User Interface

4 Multiresolution Modeling

inTouch contains a multiresolution mesh editing subsystem which takes the position of the probe on the model and the vector of applied force as input. Together, these parameters define a geometric change to the model at a certain resolution. A mesh edit is a sequence of these geometric changes applied to the model.

The user is able to edit in two different probe constraint modes and is able to choose the resolution at which the edit takes place. Recall that the highest resolution model is the one being displayed at all times. The user is free to push or pull on the surface using the probe in order to deform the geometry into a desired shape. Both the graphical and the haptic displays are updated in real-time to reflect the changing polygonal mesh.

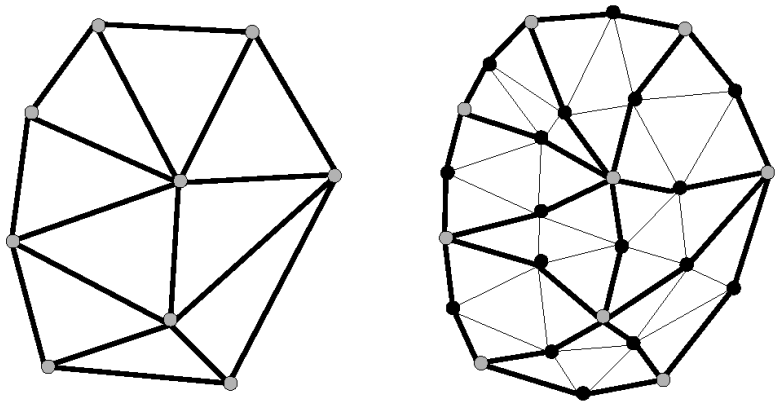

Figure 4: Triangular Subdivision Connectivity

4.1 Subdivision Surface Representation

Subdivision polygonal meshes are used as the internal representation. Input meshes without subdivision connectivity can be re-meshed using algorithms in [EDD$^+$95, KL96].

We use a triangle based type of subdivision called Loop subdivision which was invented by Charles Loop [Loop87]. It is an approximating scheme since all the vertices are repositioned each time the mesh is refined. At each stage of subdivision, each triangle is split into four smaller triangles as shown in Figure 4. Notice that along each of the edges of the coarser level, a single vertex is introduced. These new vertices are called *odd* vertices. The vertices that already existed at the coarser level are called *even* vertices.

The subdivision rules, also called stencil weights or masks, are based on the *three-directional box spline*. They are shown in Figure 5. A limit surface is the surface that is produced after an infinite number of subdivisions. The limit surface that the Loop rules produce is C^2-continuous except at extraordinary vertices (not valence 6) where there is C^1-continuity. More details on subdivision in general and Loop subdivision in particular can be found in [SZ98, Loop87].

The work of Zorin et al. [ZSS97] strongly influenced the multiresolution design of our mesh editing subsystem. We use the Loop subdivision method as described above for computing high resolution geometry from low resolution geometry. A Gaussian-like smoother proposed by Taubin [Tau95] is used to propagate geometric changes in the other direction. The smoother is a signal processing operator that resembles the Loop subdivision rules but in the reverse sense. Sharp features

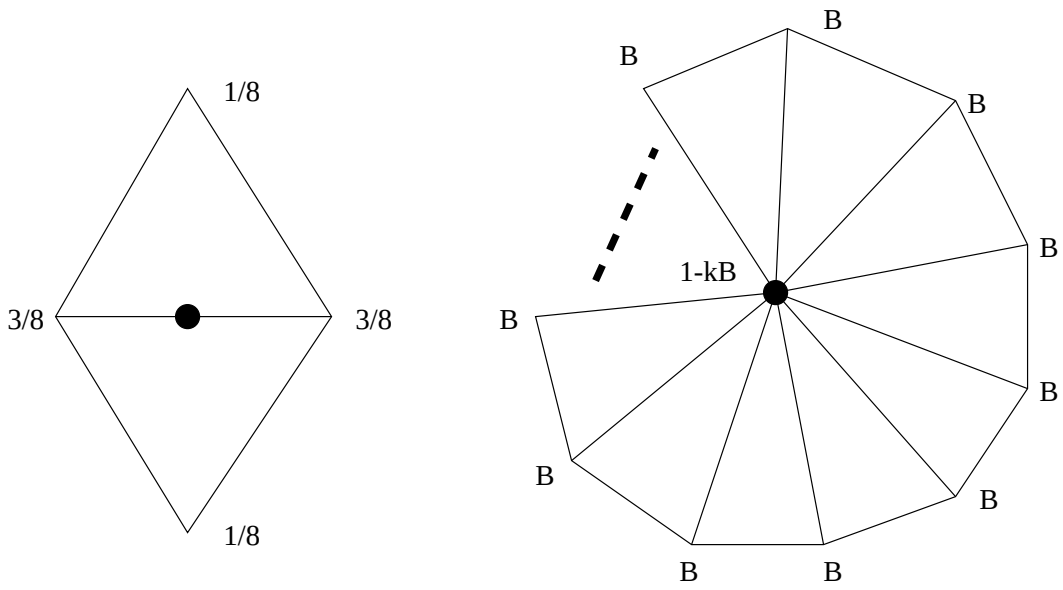

Figure 5: Loop Subdivision Rules: On the left is the rule for the odd (new) vertices and on the right is the rule for the even vertices where k is the valence of the vertex and Loop proposes that $B = \frac{1}{k}(\frac{5}{8} - (\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{k})^2)$

such as boundaries and surface derivative discontinuities can be handled through straightforward modifications of the stencil weights as shown in [HDD+94, Sch96].

Vertex positions and detail vectors are stored at each resolution level. Levels are numbered higher with increasing resolution. The detail vectors at level i represent how much the positions at level i differ from the subdivision of level $i-1$. To obtain correct editing semantics, Forsey and Bartels [FB88] have shown that the finer level details must be expressed relative to a local coordinate frame induced by the coarser level. This is also known as coordinate independence. The details are therefore expressed relative to a local frame.

When an edit occurs at a certain resolution, a set of the vertices at that level is moved. These moving vertices cause a local region of the highest resolution mesh to be affected. The displays show the highest resolution mesh, and are updated in real-time by repeatedly subdividing the affected parts of the mesh and adding in the details present at each level. Therefore, the levels finer than the edit resolution are updated using subdivision rules during the edit itself. The resulting updated finest mesh is then passed to the collision detection library, H-Collide [GLGT99], and to the graphical display. After the edit is done, the levels coarser than the edit resolution are updated using smoothing and new vertex positions and detail vectors are computed for the coarser levels.

4.2 Deforming the Model Surface

Given the subdivision framework as described above, along with the user's input, we need to decide which vertices to move and how to move them (i.e. determine how the model surface deforms due to the movement applied by the user). H-Collide provides the mesh editing subsystem with a triangle, a point of contact, and a movement vector. The movement vector $\vec{m}$ has magnitude directly related to the magnitude of the force applied by the user and direction equal to the direction of the force. A triangle existing at the edit resolution is found such that the provided triangle at the viewing (highest) resolution was derived from it. In other words, the parent of the provided triangle at the

edit resolution is found. A distribution of the movement vector must be applied across the parent triangle. We have found that the following heuristic works well in practice. The distances d_0, d_1, and d_2 from the point of contact to the vertices v_0, v_1, and v_2 of the parent triangle are respectively computed. Each of the vertices of this triangle is then moved by a (possibly) different amount $\vec{m}_i$ given by:

$$\vec{m}_i = (1 - \frac{d_i}{d_0 + d_1 + d_2})\vec{m}$$

It is possible to construct other movement distribution functions. Such a function should give more movement to nearer vertices, the movement distribution should be based on distance, and the subdivision scheme should be taken into account. Moreover, movement of vertices other than those belonging to the parent triangle should also be considered. Physically-based deformation schemes are also worthy of further investigation.

4.3 Probe Constraint Modes

The first probe constraint mode is *slide mode* which constrains the motion of the probe to lie on the model's surface. This mode is useful for "digging trenches" or "raising ridges". If the probe has moved from inside the surface to outside, then it is pulled back inside. The result of this type of modeling action allows the probe to freely move on the surface without leaving it. It can be rather difficult to push in high peaks or pull out deep indentations with this technique, because the probe will tend to slide along the surface rather than move the desired feature. For this reason we also have the second probe constraint mode called *stick mode* which attaches the probe to a point on the surface. This mode constrains the probe to a single point on the model. It is useful for creating "bumps" or "indentations". The barycentric coordinates of the surface contact point relative to its triangle at the start of the deformation are stored and at each frame of user input until the end of the deformation those coordinates are used to determine the contact point. Note that in either of these two cases, the probe is not constrained such that the user cannot move it in any direction. It is merely "attached" to the model differently in each of the two modes.

4.4 Haptic Display with Deformations

Certain extensions to H-Collide were necessary in order to allow for real-time shape change. Typically for haptic display one needs to know if the position of the probe between the last frame and the current frame has passed through the surface in order compute a surface contact point. However, since the user would like to make indentations and bumps, contacts must also be detected for the time-varying surface which exists during an editing deformation.

Pushing on the surface does not present any problems since the probe is always in contact with the surface. Pulling, on the other hand, presents a problem because as soon as the user starts pulling, the probe detaches from the surface and there is no contact point. To allow pulling, we first update the mesh with the geometric change. Then the tip of the probe is moved to a contacting position backwards along its line of movement until it is once again in contact with the surface. This keeps the probe in contact with the surface for the duration of a pulling deformation.

5 3D Painting

inTouch allows an artist or designer to paint directly onto the surface. Similar to [JTK+99], we use 3-dof haptic display to facilitate this process. However, whereas [JTK+99] used a NURBS

representation, our system allows the user to paint onto an arbitrary polygonal mesh. H-Collide is used to display the mesh and establish the point of contact of the PHANToM probe with the surface of the object. The probe is then used as a virtual paintbrush with the user's preferred brush size, color, and falloff. The brush size is stretched relative to the amount of force being applied by the stylus. This is similar to the manner in which a real paintbrush applies more paint to the surface the harder it is pressed against the surface.

5.1 Applying Paint: The Brush Function

When painting the surface, each update to the probe's location is added to the current brush stroke. Let R_p be the distance from the closest point on the stroke to the point P on the surface to which paint is being applied. Let R_b be the radius of the stroke at that closest point. It is obtained by interpolating between the radius of the stroke at two consecutive updates while composing the stroke. We color a point P on the surface within the brush radius R_b (defined by the brush size, and the force applied to the surface by the user) according to the brush function:

$$I = ((\frac{R_p}{R_b})^f - 1)^2$$
$$C = C_b * I + C_p * (1 - I)$$

where f is a user specified falloff rate, I is the intensity of the paint at point P, C_b is the current paint color selected by the user, C_p is the color that was previously painted at the point P, and C is the resulting color that smoothly blends the two colors C_b and C_p. It is easy to substitute in a more complex and creative brush function if so desired.

5.2 Applying Paint: Representation

Now that it has been established how we define the color for a point on the surface, the question arises as to what points we are painting, and how they are displayed. Hanrahan et al. [HH90] assign colors to vertices. In order to paint in detail, this requires the mesh to be subdivided into a very large number of micropolygons. Triangles must be subdivided into texel size micropolygons for this to work. Since we would like to allow the users to interactively paint on the surface in very fine detail on triangle meshes composed of only tens of thousands of triangles, we use texture maps.

Johnson et al. [JTK+99] also used texture maps, but they simply find the perimeter of the brush and perform a flood fill in image space. This approach does not work for a brush function as explained in [HH90], since "the distortion of the brush is a complicated non-linear mapping to parameter space and cannot be easily approximated". The approach in [HH90] performs 3D painting with a mouse in screen space. The problem of mapping the brush function can be avoided by taking advantage of the hardware mapping from parameter space to screen space. However, this does not work for our system since the user is free to move the virtual paintbrush in three dimensions, expecting it to paint into the tangent space of the surface (just like a real paint brush).

inTouch avoids the problem of mapping the brush function by doing a standard scan-conversion in texture space. 2D edge equations are computed in texture space for each triangle within the brush radius. Using these edge equations and the triangle's location in 3D, a plane equation is established. This plane equation is used to increment the 3D location for each texel during the scan-conversion. Hence the brush function can be applied to each texel in the triangle based on its 3D location.

5.3 Establishing the Initial Textures

If the triangle mesh comes with texture maps already on it, then the above method works without any problem. Unfortunately, this is typically not the case. Several researchers have already suggested applying texture maps to triangle meshes [FDHF90]. Some of these techniques are automatic, but most are user assisted such as [MYV93]. The most common problem of texture mapping is to find a way of mapping a set of given textures onto a model such that they do not look distorted. Since we are creating the texture contents entirely by painting, distortion will not be a problem. The surface curvature does not matter. We must however ensure that there is a high enough texel per surface area coverage for detailed painting.

6 Prototype Demonstration

inTouch is a proof-of-concept prototype system for validating the usefulness of a haptic interface for 3D modeling and painting. More than 10 novice users with little experience in using either modeling or painting systems have been able to use this system to generate interesting painted models with little training (less than 15 minutes). We have chosen novice users as they have no biased preference or preconceived notions with regard to using any modeling or painting systems. Various models created and painted by *inTouch* are shown in Color Plates 1 and 2.

All the users were asked what features of *inTouch* they liked and which parts of the system could be further improved. Here is a brief summary:

- The haptic interface provides good tactile feedback for 3D painting on the model surface. Many users like the ease and simplicity of painting directly onto the 3D model.
- Several users like the capability to easily modify the global shape of the model, while still being able to create detailed features. This is due to the choice of subdivision surfaces for multiresolution modeling.
- For detailed modeling and painting, some novice users complained of muscle fatigue while holding up their arm for a long time during use of the PHANToM.We conjecture that this is mostly due to bad ergonomics in the design of our haptic table setup. More thought needs to go into the setup and more studies need to be conducted to minimize user fatigue.
- Though the current implementation of our modeling and painting tools are limited in their capability, a complete suite of modeling and painting tools based on this system concept seem desirable.
- Several of our users experienced some difficulty associating the haptic display with the monoscopic display, due to the lack of depth cueing in 2D screen projection. When we added 3D stereoscopic display, the problem was alleviated.

7 Summary and Future Work

inTouch provides interactive multiresolution modeling and 3D painting capabilities using an intuitive haptic interface. Haptic display overcomes limited modes of interaction provided by traditional 2D interfaces such as mice and keyboards, and allows artists and designers to freely express their creativity by a sense of touch.

There are several interesting research areas to pursue:

- Experimenting with two-handed, multi-user and/or distributed haptic interaction and conducting a thorough user study on different UI paradigms for modeling and 3D painting.
- Integrating a 6-degrees-of-freedom haptic device with the system. This would create more interesting brush functions by applied torque, and add a whole new level of usability to modeling by twisting and other special effects.
- Exploring other movement distribution functions to map the contact location and the direction of applied force to vertex movement at the edit level.
- Adding a complete artistic suite of brush functions similar to the popular 2D painting programs today. Introducing the ability to cut and paste texture images in surface space directly.
- Creating surface roughness as the result of adding paint as well as resistance when painting. Brush bristles could be modeled and force transmitted along them.

References

[ABF+97] Maneesh Agrawala, Andrew C. Beers, Bernd Fröhlich, Pat Hanrahan, Ian McDowall, and Mark Bolas. The two-user responsive workbench: Support for collaboration through independent views of a shared space. *SIGGRAPH 97 Conf. Proc.*, pages 327–332, 1997.

[ABL95] Maneesh Agrawala, Andrew C. Beers, and Marc Levoy. 3D painting on scanned surfaces. *Proc. of 1995 ACM Symposium on Interactive 3D Graphics*, pages 145–150, 1995.

[APT+98] K. Arthur, T. Preston, R. Taylor, F. Brooks, M. Whitton, and W. Wright. Designing and building the pit: A head-tracked stereo workspace for two users. *Proc. of 2nd International Immersive Projection Technology Workshop*, 1998.

[AS96] R. S. Avila and L. M. Sobierajski. A haptic interaction method for volume visualization. *Proceedings of Visualization'96*, pages 197–204, 1996.

[CB94] J. E. Colgate and J. M. Brown. Factors affecting the z-width of a haptic display. *IEEE Conference on Robotics and Automation*, pages 3205–3210, 1994.

[CFH97] L. Cutler, B. Frolich, and P. Hanrahan. Two-handed direct manipulation on the responsive workbench. *Proc. of 1997 Symposium on Interactive 3D Graphics*, pages 107–114, 1997.

[Col94] J. E. Colgate, et al. Issues in the haptic display of tool use. *Proceedings of the ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 140–144, 1994.

[Coq90] Sabine Coquillart. Extended free-form deformation: A sculpturing tool for 3D geometric modeling. *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 187–196, August 1990.

[CR94] Yu–Kuang Chang and Alyn P. Rockwood. A generalized de Casteljau approach to 3D free–Form deformation. *Proceedings of SIGGRAPH '94*, pages 257–260, July 1994.

[DKT98] T. DeRose, M. Kass, and T. Troung. Subdivision surfaces in character animation. *Computer Graphics (ACM SIGGRAPH'98)*, pages 85-94, 1998.

[EDD+95] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. *Proc. SIGGRAPH 95 Conf. Proc.*, pages 173–182, 1995.

[Far90] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide.* Academic Press Inc., 1990.

[FB88] D. Forsey and R.H. Bartels. Heirarchical b-spline refinement. In *Proc. of ACM SIGGRAPH*, pages 205–212, 1988.

[FDHF90] J. Foley, A. Van Dam, J. Hughes, and S. Feiner. *Computer Graphics: Principles and Practice.* Addison Wesley, Reading, Mass., 1990.

[FFC+95] M. Finch, M. Falvo, V. L. Chi, S. Washburn, R. M. Taylor, and R. Superfine. Surface modification tools in a virtual environment interface to a scanning probe microscope. *Proc. of ACM 1995 Symposium on Interactive 3D Graphics*, pages 13–18, 1995.

[GH91] Tinsley A. Galyean and John F. Hughes. Sculpting: An interactive volumetric modeling technique. *Proc. of Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 267–274, 1991.

[Gib95] S. Gibson. Beyond volume rendering: Visualization, haptic exploration, an d physical modeling of element-based objects. In *Proc. of Eurographics workshop on Visualization in Scientific Computing*, pages 10–24, 1995.

[GLGT99] A. Gregory, M. Lin, S. Gottschalk, and R. Taylor. H-Collide: A framework for fast and accurate collision detection for haptic interaction. In *Proceedings of IEEE Virtual Reality Conference 1999*, pages 38-45, 1999.

[GLGT00] A. Gregory, M. Lin, S. Gottschalk, and R. Taylor. Real-time collision detection for haptic interaction using a 3-dof force feedback device. In the special issue of *Computational Geometry: Theory and Applications*, Vol. 15, No. 1-3, pages 69-89, February 2000.

[HDD+94] Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise smooth surface reconstruction. *Proc. of SIGGRAPH '94*, pages 295–302, 1994.

[He97] J. Hollerback and et al. Haptic rendering for virtual prototyping of mechanical cad designs. *Proc. Design for Manufacturing Symposium*, 1997.

[HH90] Pat Hanrahan and Paul E. Haeberli. Direct WYSIWYG painting and texturing on 3D shapes. *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 215–223, 1990.

[HHK92] William M. Hsu, John F. Hughes, and Henry Kaufman. Direct manipulation of free-form deformations. *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 177–184, 1992.

[JC98] D. Johnson and E. Cohen. A framework for efficient minimum distance computation. *IEEE Conference on Robotics and Automation*, pages 3678–3683, 1998.

[JTK+99] D. Johnson, T. V. Thompson II, M. Kaplan, D. Nelson, and E. Cohen. Painting textures with a haptic interface. *Proc. of IEEE Virtual Reality Conference*, 1999.

[KL96] Venkat Krishnamurthy and Marc Levoy. Fitting smooth surfaces to dense polygon meshes. *SIGGRAPH 96 Conference Proceedings*, pages 313–324, 1996.

[KS99] A. Khodakovsky and P. Schröder. Fine level feature editing for subdivision surfaces. *Proc. of ACM Symposium on Solid Modeling and Applications*, 1999.

[Loop87] C. Loop. Smooth subdivision surfaces based on triangles. M.S. Thesis, University of Utah, Department of Mathematics, 1987.

[Mas98] Thomas Massive. A tangible goal for 3d modeling. *IEEE Computer Graphics and Applications*, May/June, pp. 62-65, 1998.

[MBS97] Mark R. Mine, Frederick P. Brooks, Jr., and Carlo H. Séquin. Moving objects in space: Exploiting proprioception in virtual-environment interaction. *SIGGRAPH 97 Conf. Proc.*, pages 19–26, 1997.

[MJ96] Ron MacCracken and Kenneth I. Joy. Free-Form deformations with lattices of arbitrary topology. *SIGGRAPH 96 Conf. Proc.*, pages 181–188, 1996.

[MRF+96] William Mark, Scott Randolph, Mark Finch, James Van Verth, and Russell M. Taylor II. Adding force feedback to graphics systems: Issues and solutions. *SIGGRAPH 96 Conf. Proc.*, pages 447–452, 1996.

[MS94] T. M. Massie and J. K. Salisbury. The phantom haptic interface: A device for probing virtual objects. *Proc. of ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 1:295–301, 1994.

[MYV93] Jérôme Maillot, Hussein Yahia, and Anne Verroust. Interactive texture mapping. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 27–34, August 1993.

[NNHJ98] A. Nahvi, D. Nelson, J. Hollerbach, and D. Johnson. Haptic manipulation of virtual mechanisms from mechanical cad designs. In *Proc. of IEEE Conference on Robotics and Automation*, pages 375–380, 1998.

[OY90] M. Ouh-Young. *Force Display in Molecular Docking.* PhD thesis, University of North Carolina, Computer Science Department, 1990.

[PBBW95] Randy Pausch, Tommy Burnette, Dan Brockway, and Michael E. Weiblen. Navigation and locomotion in virtual worlds via flight into Hand-Held miniatures. *SIGGRAPH 95 Conference Proceedings*, pages 399–400, 1995.

[PFC+97] J. Pierce, A. Forsberg, M. Conway, S. Hong, R. Zeleznik, and M. Mine. Image plane interaction techniques in 3d immersive environments. *Proc. of 1997 Symposium on Interactive 3D Graphics*, pages 39–44, 1997.

[PT97] L. A. Piegl and W. Tiller. *The NURBS Book.* Springer Verlag, 1997. 2nd Edition.

[QT96] Hong Qin and Demetri Terzopoulos. D-NURBS: A physics-Based framework for geometric design. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):85–96, March 1996.

[RE99] A. Raviv and G. Elber. Three dimensional freeform sculpting via zero sets of scalar trivariate functions. *Proceedings of Symposium on Solid Modeling and Applications*, 1999.

[RH92] Warren Robinett and Richard Holloway. Implementation of flying, scaling, and grabbing in virtual worlds. *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, volume 25, pages 189–192, March 1992.

[RKK97] D.C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. *Proc. of ACM SIGGRAPH*, pages 345–352, 1997.

[Sch96] J. E. Schweitzer. *Analysis and Application of Subdivision Surfaces.* PhD thesis, University of Washington, 1996.

[SP86] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 151–160, 1986.

[ST99] SensAble Technologies Inc. $freeform^{TM}$ modeling system. *http://www.sensable.com/freeform*, 1999.

[SZ98] P. Schröder and D. Zorin. Subdivision for modeling and animation. *ACM SIGGRAPH Course Notes*, 1998.

[SZMS98] T. Sederberg, J. Zheng, M.Sabin, and D. Sewell. Non-uniform recursive subdivision surfaces. *Computer Graphics (ACM SIGGRAPH'98)*, 1998.

[Tau95] Gabriel Taubin. A signal processing approach to fair surface design. *SIGGRAPH 95 Conf. Proc.*, pages 351–358, 1995.

[TRC+93] R. M. Taylor, W. Robinett, V.L. Chii, F. Brooks, and W. Wright. The nanomanipulator: A virtual-reality interface for a scanning tunneling microscope. In *Proc. of ACM Siggraph*, pages 127–134, 1993.

[VRPN] Virtual Reality Peripheral Network. http://www.cs.unc.edu/research/nano/manual/vrpn.

[ZSS97] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. *Computer Graphics (ACM SIGGRAPH'97)*, 1997.