

Andrew J. Doxon*
Department of Mechanical
Engineering
University of Utah
Salt Lake City, Utah 84112

David E. Johnson
Department of Computer Science
University of Utah
Salt Lake City, Utah 84112

Hong Z. Tan
Haptic Interface Research
Laboratory
Purdue University
West Lafayette, Indiana 47907

William R. Provancher
Department of Mechanical
Engineering
University of Utah
Salt Lake City, Utah 84112

Force and Contact Location Shading Methods for Use Within Two- and Three-Dimensional Polygonal Environments

Abstract

Current state-of-the-art haptic interfaces only provide kinesthetic (force) feedback, yet studies have shown that providing tactile feedback in concert with kinesthetic information can dramatically improve a person's ability to dexterously interact with and explore virtual environments. In this research, tactile feedback was provided by a device, called a contact location display (CLD), which is capable of rendering the center of contact to a user. The chief goal of the present work was to develop algorithms that allow the CLD to be used with polygonal geometric models, and to do this without the resulting contact location feedback being overwhelmed by the perception of polygonal edges and vertices. Two haptic shading algorithms were developed to address this issue and successfully extend the use of the CLD to 2D and 3D polygonal environments. Two experiments were run to evaluate these haptic shading algorithms. The first measured perception thresholds for rendering faceted objects as smooth objects. It was found that the addition of contact location feedback significantly increased user sensitivity to edges and that the use of shading algorithms was able to significantly reduce the number of polygons needed for objects to feel smooth. The second experiment explored the CLD device's ability to facilitate exploration and shape recognition within a 3D environment. While this study provided a validation of our 3D algorithm, as people were able to identify the rendered objects with reasonable accuracy, this study underscored the need for improvements in the CLD device design in order to be effectively used in general 3D environments.

I Introduction

Human-computer interfaces that involve the sense of touch, or haptic interfaces, are becoming more and more prevalent throughout the world. Despite this, these devices are still often restrictive and frustrating to use, which keeps them far from their full potential as intuitive human-computer interfaces.

Most current haptic interfaces provide a purely kinesthetic interaction within virtual environments. This results in a significant loss of dexterity, as reported by Frisoli, Bergamasco, Wu, and Ruffaldi (2005). If implemented well, providing tactile feedback in combination with kinesthetic information should dramatically improve one's ability to dexterously interact and explore virtual environments, to potentially provide an improvement similar to when people remove a pair of gloves.

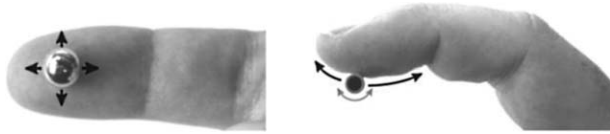


Figure 1. Concept for contact location feedback. The (left) two-dimensional or (right) one-dimensional center of contact is represented with a single tactile element.

One such system that provides both tactile and kinesthetic feedback is the contact location display (CLD) developed by Provancher, Cutkosky, Kuchenbecker, and Niemeyer (2005) attached to a PHANToM. In addition to forces, this device displays the contact location between a virtual finger and a surface to the user. Figure 1 shows the concept for a contact location display.

Previously, the CLD device was utilized only with specialized 2D models. Use of 3D polygonal geometric models, as is common in both haptics and computer graphics (Ruspini & Khatib, 2001), with the CLD device would significantly expand the device's usefulness by allowing combined tactile and kinesthetic feedback in these common virtual environments without requiring model conversion or preprocessing.

However, when interacting with polygonal approximations to smooth surfaces, the CLD transmits the surface discontinuities to the user. This gives the impression that the surface is meant to be rough or textured rather than smooth and it is distracting to the user even when interacting with high-count polygonal models. The use of shading algorithms could potentially not only reduce the effects of surface discontinuities, but also lead to a significant reduction in model size while still retaining a surface that feels smooth.

Force shading, as developed by Morganbesser and Srinivasan (1996), smoothes the faceted models by interpolating the surface normal between vertices. Discontinuities in the form of proprioceptive (position) cues remain present. Humans, in general, find it difficult to detect these proprioceptive cues so the smooth force interactions override the weaker proprioceptive signals and a smooth object is perceived. However, contact location is dependent on the object's surface and the virtual finger, which are not altered by Morganbesser and Srinivasan's force shading. The state-of-the-art is therefore incapable

of eliminating the discontinuities in the tactile feedback for the CLD device (and other tactile displays).

This paper presents two related haptic shading algorithms to provide smooth tactile and kinesthetic feedback for use within general 2D and 3D polygonal environments. These algorithms are designed to provide only a single point of contact, matching the display capabilities of the CLD device, and function on both convex and concave surfaces. The 2D shading algorithm was developed, implemented, and tested with human subjects to determine the feasibility of our approach and to obtain perceptual thresholds for rendering smooth objects. A more advanced 3D algorithm was then developed and tested using the results from the first experiment. The algorithms each derive locally smooth feedback from the original polygonal model. Some advantages of the algorithm include improved kinesthetic display over just using force shading and smoothed contact location feedback in the presence of polygonal artifacts. Furthermore, our approach is computationally efficient, making smoothed interactions feasible with complex environments and arbitrary finger models.

Section 2 provides a brief background concerning the literature most relevant to this research. Section 3 is a description of the CLD device. The 3D algorithm is then presented in detail in Section 4, with details of the 2D algorithm presented in the Appendix. In Section 5, we present two human subject experiments. The first experiment establishes the necessary polygonal mesh parameters for shaded polygon objects to feel perceptibly smooth. The second experiment is an object identification task, which provides a validation of the developed 3D algorithm and provides insights and inspiration for future work to further improve the efficacy of contact location feedback. We present conclusions in Section 6.

2 Background

2.1 Combined Tactile and Kinesthetic Feedback

A number of studies have been conducted with combined tactile and kinesthetic feedback. Salada, Colgate, Vishton, and Frankel (2005) conducted several

studies that investigated the use of slip or sliding feedback in combination with kinesthetic motions. Salada was able to show that the addition of slip feedback allowed users to track small moving features better. The saliency of friction is also increased with skin stretch and slip feedback (Provancher & Sylvester, 2009). Since then, others have also developed slip displays and integrated them with kinesthetic force feedback devices (Fritschi, Ernst, & Buss, 2006; Webster, Murphy, Verner, & Okamura, 2005). These devices tend to be large and cumbersome since a smaller contact area on the finger relates to weaker sliding cues. Fritschi et al. found that users judged interactions with slip feedback as more real. Additionally, they also investigated providing tactile slip feedback with a tactile pin array in combination with kinesthetic feedback. Again, Fritschi et al. found that providing slip feedback from a pin array increased the realism of the models. Like slip displays, pin arrays tend to be large and cumbersome. However, the true benefit of pin arrays is the variety of interactions possible with the device. Each pin can be individually controlled to create the sensation of textures across virtual surfaces.

Other interesting approaches to tactile-kinesthetic display include research on displaying the local object surface tangent (Dostmohamed & Hayward, 2005; Frisoli, Solazzi, Salsedo, & Bergamasco, 2008). Dostmohamed and Hayward present a device that utilizes a gimbaled plate to represent the local surface tangent plane of virtual objects. The motion of the gimbaled plate is coordinated with the user's kinesthetic motion to display curved objects. Dostmohamed and Hayward were able to demonstrate that by providing only an object's tangent plane through a gimbaled plate, participants were capable of curvature discrimination on a par with real-life exploration of large objects. As a relatively sophisticated adaptation of this work, Frisoli et al. present a miniaturized finger-based tilting plate tactile display that can be attached to a kinesthetic display. Their results indicate a significantly improved performance in curvature discrimination when kinesthetic cues are also given.

Finally, Provancher's prior studies have shown the potential of contact location feedback for enhancing

object curvature and motion cues (Provancher et al., 2005). The CLD has been shown to increase awareness of curvature change and edges, which enables better contour following (Kuchenbecker, Provancher, Niemeyer, & Cutkosky, 2004).

2.2 Haptic Shading Algorithms

Haptic shading algorithms are developed to make polygonal representations of smooth objects feel smooth. Without haptic shading algorithms, polygonal models of smooth objects feel rough and textured, which detracts from the desired haptic experience. Most shading algorithms either directly modify the interaction with the polygonal model or alter the position of a virtual proxy, a copy of the virtual finger left on the model's surface to which forces are rendered.

The most widely used haptic shading algorithm was developed by Morganbesser and Srinivasan (1996). This algorithm linearly interpolates surface normals on the environment models to guarantee a continuously smooth gradient. The graphics community uses a similar technique called Phong shading to create smooth normals for evaluating illumination across polygonal surfaces (Phong, 1973). Morganbesser and Srinivasan's algorithm was designed to reduce the popping effect felt in rendered normal forces when the haptic interaction point passes over a vertex or edge of a polygonal object. As with Phong shading, Morganbesser and Srinivasan found that their force shading algorithm helped give the sensation of a smoother object.

Ruspini, Kolarov, and Khatib (1997) also incorporated a force shading model which interpolates the normals of the surface. In this case, a two-pass technique was utilized to modify the position of the virtual proxy. The first stage computes the closest point on the plane defined by the interpolated normal and the current proxy position. The second stage computes proxy forces as usual but uses the previously found closest point as the user-controlled point. This method reduces instability issues generated by using the original Morganbesser and Srinivasan algorithm when the haptic interaction point is in contact with multiple intersecting shaded surfaces.

An alternative to shading polygonal surfaces is to work directly with NURBS (nonuniform rational B-spline surface) models. Rather than approximating a surface, NURBS models use piecewise rational surfaces with controllable smoothness to precisely represent shapes. Existing approaches for haptic rendering of these models exploit tracking of a local contact point on the model (Thompson & Cohen, 1999; Johnson & Cohen, 1999) and between two models. However, creation of detailed NURBS models is still a complex task, and conversion from arbitrary models with complex topologies is even more so. This paper provides a direct means of haptic interaction with polygonal mesh surface models while retaining some of the tracking and surface smoothness properties algorithms for NURBS models.

Other model representations, like the voxel approach presented by McNeely, Puterbaugh, and Troy (1999), include haptic shading through the summation of each voxel in the modeled environment. In this way, small motions create small changes across multiple voxels, thus creating the effect of a smooth interaction. However, methods like these provide only forces and cannot provide a contact location to be rendered with the CLD.

3 Experimental Apparatus

The concept for contact location feedback is presented in Figure 1, where only the center of contact is rendered. The hardware utilized in the following experiments consists of a SensAble PHANToM Premium 1.5, and a 1-DOF CLD device which displays contacts along the finger (see Figures 2 and 3). The PHANToM is used to render contact forces. The contact location display is used to render the current contact position on the finger. The device utilizes a 1 cm diameter delrin roller as a tactile contact element. The position of the roller on the finger is actuated via sheathed push-pull wires attached to a linear actuator mounted on the user's forearm. The display's contact roller is directly attached to the PHANToM via a 1-DOF gimbal with a sensed tilt angle. The roller is suspended beneath the fingerpad by the drive wires so that it does not touch the user's finger until contact is made with a virtual object. Contact forces, provided by the PHANToM, push the roller into contact

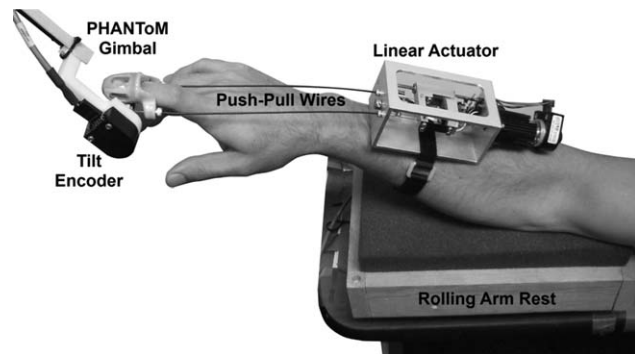


Figure 2. Contact location display prototype attached to a PHANToM robot arm. The user's elbow is supported by a rolling armrest.

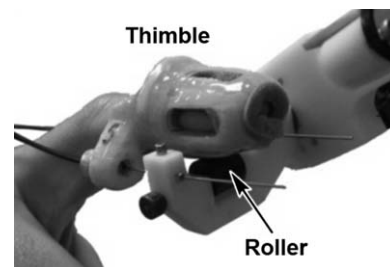


Figure 3. The user's finger is secured to the contact location display via an open-bottom thimble.

with the user's fingerpad. An open-bottom thimble is used to attach the device securely to a user's finger and also provides a mounting point to anchor the sheaths of the spring steel drive wires. Several interchangeable thimbles, which together accommodate a wide range of finger sizes, were created using fused deposition modeling (FDM) rapid prototyping.

The linear actuator is located on the user's forearm to prevent any possible device vibrations from being transmitted to the user's fingertip receptors and to reduce the device inertia located at the fingertip. The linear actuator utilizes a Faulhaber 2342CR DC brushed motor and a 3.175 mm pitch lead screw to provide approximately 2 cm of linear motion with approximately 0.8 μm of resolution and a bandwidth in excess of 5 Hz. A prototype of the device can be seen in Figure 2. A close-up view of the fingertip portion of the device is shown in Figure 3.

The device's motor is driven by an AMC 12A8 PWM amplifier that is controlled using a Sensoray 626 PCI control card. The device's PID controller was run at

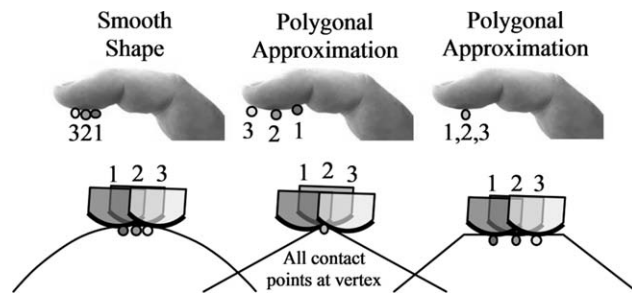


Figure 4. Contact location movement over a smooth round surface represented (left) with a curved surface model, (middle) with two facets, and (right) with three facets. The top shows a view of the fingerpad with a series of displayed contact locations, corresponding by shade and number to the virtual finger positions below.

1 kHz and was programmed in C++. The control program was executed under Windows XP using Windows multimedia timers. Further details about the design and control of this device may be found in Provancher et al. (2005).

4 Contact Location Rendering and Haptic Shading

4.1 Smooth versus Faceted Surfaces

Many models in virtual environments are composed of faceted triangle meshes, even when the desired shape is smooth and continuous. In order to facilitate the use of tactile feedback during manipulation, the original smooth shape must be recovered. Without smoothing, the edges of the triangle mesh dominate all other tactile information provided by the CLD.

The motion of the CLD device depends on the shape of the model used. The tactile motion of the CLD device traveling over a smooth curve in comparison to faceted surfaces is demonstrated in Figure 4. Note that the contact location smoothly changes while moving along a curved surface, whereas the contact location moves rapidly along the finger when crossing a vertex, and remains stationary while traversing a flat facet.

The following sections describe our algorithms to recover a smoothed version of a faceted model and to use this smoothed surface to render appropriate kinesthetic and tactile cues during contact.

4.2 Overview of Developed Algorithms

Both the 2D and 3D smoothing algorithms presented in this paper utilize Bézier curves/surfaces to generate smooth interactions. These curves/surfaces are temporarily generated from a control polygon produced from the underlying environment model around the region of contact. The resulting Bézier curve/surface is then used with the finger model to determine the proper contact location and force feedback parameters. This approach is a hybrid of prior work on rendering and shading triangular mesh models and work on rendering parametric models, such as splines, as it works directly with the given polygonal models yet locally generates a temporary parametric surface for smoothing.

In general, computing the contact location between two curves, the finger model and curved environment model, requires robust numerical methods that may run too slowly for haptic applications (Seong, Johnson, Elber, & Cohen, 2010). Instead, our algorithm computes a dynamically updated tangent line/plane at the point of contact. This reduces the computation needed to evaluate the interaction between a line/plane and the finger model. This interaction is rendered as a single point that is constrained to lie on the finger model's surface, which matches the display capabilities of the CLD. Thus, the approach is not based purely on a point-model or model-model interaction, but instead lies somewhere in between.

By ensuring the environment model is a fully connected and continuous manifold mesh, we can guarantee the resulting curve/surface is continuous and smooth. Multiplicity, or multiple points/normals defined at the same coordinates, can be used to generate sharp corners on the rendered smooth surface when desired.

In brief, the algorithms perform the following steps:

1. First the model is broken into a local control polygon/mesh.
2. The contact location with the current tangent line/plane is computed to evaluate finger motion with respect to the surface.
3. Given the local control polygon/mesh and motion along the tangent, the motion along the smoothed

surface is approximated and a new tangent line/plane is computed.

4. This approximation iterates until convergence with the true contact location is reached.
5. The final tangent after convergence is then used to compute the displacement of the CLD as well as the smoothed forces rendered by the PHANTOM.

While the algorithm was developed to provide both smooth tactile and kinesthetic feedback, it can also be used as a substitute for the methods presented by Morganbesser and Srinivasan (1996) for force shading.

Details of the 3D haptic shading algorithm are presented below, while the 2D algorithm that was used in our discrimination threshold study is included, for completeness, in the Appendix.

4.3 Overview of the 3D Haptic Shading Algorithm

Each primitive triangle element of the polygonal model is used to generate the control mesh of a curved surface, in this case with a variant of Bézier triangles which provides contact continuity and smoothness. While it is possible to fit smooth surfaces to polygonal models, the process is difficult and time-consuming (Cohen, Riesenfeld, & Elber, 2001; Daniels, Silva, Shepherd, & Cohen, 2008).

We adapted a technique from the computer graphics literature, PN (point normal) triangles (Vlachos, Peters, Boyd, & Mitchell, 2001), which produce control meshes for Bézier triangles and quadratic interpolation based solely on the original triangle vertices and their corresponding normal vectors. This allows PN triangles to perform local smoothing processes independently of the number of triangles in the mesh which is necessary for smoothing large models at haptic rates. Evaluation of PN triangles directly defines the tangent plane used in the haptic rendering algorithm. The Bézier triangle surface provides the point and the quadratically interpolated normals provide the normal.

The process followed by the 3D algorithm uses numerical methods to converge to the ideal contact point. Thus, within each haptic rendering cycle (a minimum of

1,000 Hz) this process is repeated until the ideal contact point is reached, that is, until the proxy's contact location is the point generated by the Bézier triangle surface. The user's movement is only captured once each haptic rendering cycle. To facilitate fast rendering times, each triangle in the mesh also contains information on its three adjacent triangles.

Fully smoothed surfaces can lose important detail. The presented approach allows preservation of straight edges through the addition of multiple normals on a single vertex. These normals must be defined perpendicular to the straight edge or the PN triangle surface will become discontinuous, creating a hole in the surface. For curved edges, it is advised instead to add smaller triangles along the edge to more accurately define the feature.

4.4 PN Triangles

4.4.1 Defining the Control Mesh. PN triangles use barycentric coordinates, which are commonly used to define positions on triangles in terms of u , v , and w , as parametric coordinates. They are a system of homogeneous coordinates based on the signed areas of the base triangle and the subtriangles formed by the target point.

The Bézier triangle's control mesh in PN triangles is defined by 10 points. This creates a third order surface in all three barycentric coordinates (u , v , and w). Third order surfaces were chosen because they are the minimum degree capable of rendering inflections in surface contours. The control mesh is computed from the base triangle's points (P_1 , P_2 , P_3) and their corresponding normals (N_1 , N_2 , N_3). For the specific method of computing all 10 control points, the reader is referred to Vlachos et al. (2001).

Each edge of the control mesh is determined only by the two points comprising that edge. Thus the edges of two adjacent PN triangles are contiguous. Figure 5 shows a shaded base triangle and its corresponding control mesh. The three outermost triangles are created such that they share the base triangle's corners and normals. The center point, b_{111} , is defined as an extension of the six new middle points with respect to the original center of the base triangle.

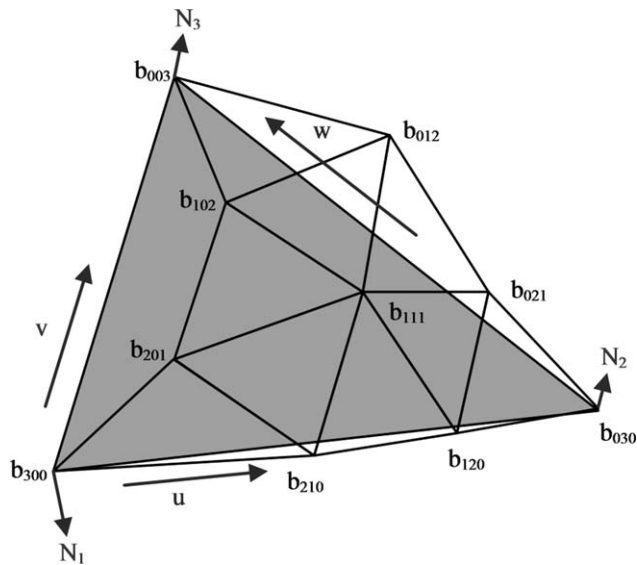


Figure 5. A control mesh generated for a particular base polygon. The mesh is defined completely by the three normals defined at each of the three vertices on the base polygon and their relationships. Arrow vectors show the directions of the barycentric coordinates u , v , and w used as parametric inputs.

The naming convention chosen in this paper is the same as that used by Vlachos et al. (2001). The base indices on the mesh represent the position and weight of each corner of the base triangle on the individual point. Thus, in Figure 5, the index of b_{012} indicates it is influenced proportionally by $0/3$ of P_1 , $1/3$ of P_2 , and $2/3$ of P_3 . The weights always sum to the order of the system being made.

The control mesh for the quadratically interpolated normals contains only six points and defines a second order system. The specific equations used by Vlachos et al. (2001) help guarantee that if there is an inflection in the surface, it will also be represented in the normals. Since this control mesh is constructed of normal vectors, all its vectors must be normalized to 1 before being used. The second control mesh uses the same naming scheme as the first (e.g., n_{110}). Since it is second order, the weights will sum to 2.

4.4.2 Computing the PN Surface. Given the control meshes and a set of barycentric coordinates, a point and normal on the surface can be computed.

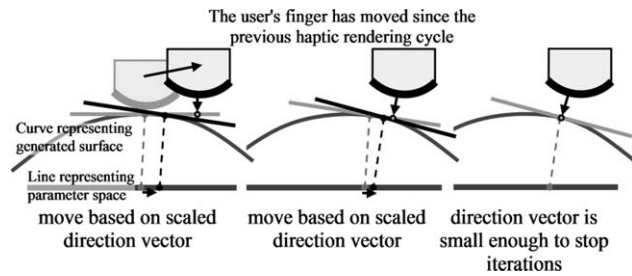


Figure 6. The tangent plane converges to the ideal contact point where the proxy contact point is the rendered surface point and is drawn chronologically from the left to the right. The previous iteration is shown in gray.

Equations are provided by Vlachos et al. (2001) to directly compute a single point and normal on the PN triangle surface. Using these continuous surface points and normals, we can guarantee that the resulting contact location will also be continuous. While a method for recursively computing the surface point and normal does exist for Bézier triangles (as the one used in the 2D algorithm), it is faster to compute the result directly in this case.

4.5 Implementing the 3D Haptic Shading Algorithm

This section provides detailed descriptions of each step taken in the algorithm. As the user moves, the shading algorithm computes a point and tangent plane on the smoothed surface.

Figure 6 demonstrates the basic iterative process performed by the shading algorithm for a typical 2D cross-section of a shaded surface. The user's finger is orthogonally projected onto the previous iteration's tangent plane (shown in gray) to compute a contact position. This contact position is used to compute the current tangent plane (shown in black). This is repeated until the tangent planes become nearly identical. The final tangent plane is then used to compute the haptic interaction.

4.5.1 Computing the Current Proxy Contact Location. In this step, the updated position of the user is orthogonally projected toward the tangent plane created in the previous iteration. The initial contact

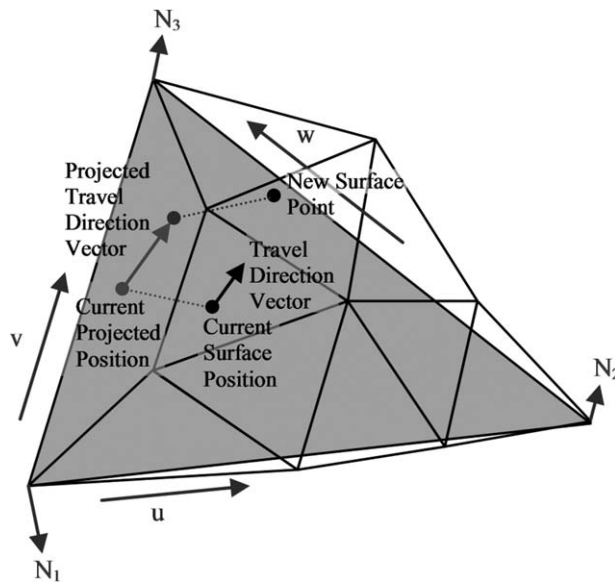


Figure 7. The travel direction vector is computed based on the current surface position. The projected direction vector is applied to the corresponding current position point on the base triangle. The resulting point is then used to compute the new surface point. Dashed lines denote a connection between the points on the base triangle and the curved surface from the Bézier control polygon.

between the finger model and tangent plane defines a new contact position. A direction vector can then be created between the previous contact position and the current contact point. This direction vector represents a reasonable linear approximation of the motion along the base triangle needed to compute the barycentric coordinates that will result in a more accurate surface rendering, thus allowing the system to converge given a sufficiently small step size.

4.5.2 Computing the New Parameter

Value. The direction vector found in the previous step is used to compute a new set of barycentric values by projecting it onto the plane of the base triangle. Figure 7 shows the travel direction vector, its projection, and the new surface point due to that projection.

Since the direction vector describes a linear approximation to the motion along the base triangle, it becomes worse with increasing curvature which is compounded by distance from the surface. Therefore, to improve stability on a wider range of surfaces while keeping the con-

vergence times small, a gain based on curvature and distance from the surface was used to minimize overshoot when estimating a more accurate contact location. The inclusion of this gain substantially improves the stability and convergence of the system across a variety of object models. Equation 1 shows the computation of this gain where G_o is the overall gain, k is the curvature in the direction of travel, d is the finger's distance from the tangent plane, and G_k and G_d are positive factors relating the importance of curvature and distance respectively when computing the next iteration's position. It should be kept in mind that increasing G_k and G_d to increase stability for high curvature models also increases convergence time and thus limits the maximum haptic rate.

$$\text{Gain} = \frac{G_o}{(1 + G_k k)(1 + G_d |d|)}. \quad (1)$$

The distance from the surface (d) is defined as the distance from the current position of the user to the proxy model in contact with the tangent plane. Since arc length increases linearly with radius, the further the user is from the surface, the smaller the angle change needed to align the normal with a particular movement. Thus, the gain is reduced linearly by the distance from the surface to ensure that smaller parametric steps are taken.

The curvature (k) is the directional curvature of the surface which is based on the curve formed by intersecting the surface with a normal plane in the travel direction. Since this space curve is not usually in the general arc length parameterized form, the most basic definition of curvature is the magnitude of the rate change of the tangent vector divided by the rate change of position along the curve (see Equation 2). Since the normals defined for each point are not the normals of the Bézier triangle surface, this equation in its pure form cannot be used. However, since by definition, on noncomposite surfaces, the normal vector (\dot{N}) and the tangent vector (\dot{T}) are always orthogonal, the magnitudes of their derivatives are also equal. Because we have separate equations for the position (s) and normal vector (\dot{N}) using barycentric coordinates, and are capable of computing the derivative of each, the final equation used to compute the curvature (k) for our composite surface is the

magnitude of the rate change of the normal (N) divided by the rate change in position (s).

$$k = \frac{\|dT\|}{\|ds\|} = \frac{\|dN\|}{\|ds\|}. \quad (2)$$

The derivatives that define curvature (dN and ds) are relatively simple to compute using the chain rule (see Equations 3–8). Each of these equations is intended to produce a value used in the following equations, leading eventually to dN and ds . Since barycentric coordinates are homogeneous ($u + v + w = 1$), only two variables (commonly u and v) are needed to define the system. Depending on the major component of the direction vector, from (u_1, v_1) to (u_2, v_2) , one of the two equation sets shown in Equation 3 should be used as the basic derivative to guarantee that \dot{u} and \dot{v} are bounded. The curvature in Equation 2 is scalar invariant with respect to the magnitude of the (u, v, w) derivative vector, thus both representations are equally valid.

$$\begin{aligned} \dot{v} = 1 & & \dot{u} = 1 \\ \dot{u} = \frac{u_2 - u_1}{v_2 - v_1} & & \dot{v} = \frac{v_2 - v_1}{u_2 - u_1} \\ \dot{w} = -\dot{u} - \dot{v} & & \dot{w} = -u - v. \end{aligned} \quad (3)$$

Next, the partial derivatives of position with respect to u , v , and w are computed (see Equation 4). The derivative of position with respect to the system is then computed using chain rule composition (see Equation 5). This derivative is the value of ds in Equation 2 when the current barycentric coordinates are plugged in.

$$\begin{aligned} \frac{\partial P}{\partial w} &= P_w = 3(b_{300}w^2 + b_{120}u^2 + b_{102}v^2) \\ &\quad + 6(b_{210}wu + b_{201}wv + b_{111}uv) \\ \frac{\partial P}{\partial v} &= P_v = 3(b_{003}v^2 + b_{201}w^2 + b_{021}u^2) \\ &\quad + 6(b_{102}wv + b_{012}uv + b_{111}uw) \\ \frac{\partial P}{\partial u} &= P_u = 3(b_{030}u^2 + b_{210}w^2 + b_{012}v^2) \\ &\quad + 6(b_{120}wu + b_{021}uv + b_{111}vw). \end{aligned} \quad (4)$$

$$ds = \frac{d}{d(u \text{ or } v)} P = P_w \dot{w} + P_u \dot{u} + P_v \dot{v}. \quad (5)$$

All that remains is to compute the derivative of the normal vector before curvature can be calculated. Since the normal vector is divided by its magnitude, the resulting chain form equation is slightly more complicated. Firstly, N and its derivatives are computed (see Equations 6, 7, and 8). Then the derivative of the unit normal can be computed using Equation 8. Finally, the current barycentric coordinates and Equations 5 and 8 are used to compute the curvature (see Equation 2).

$$\begin{aligned} \frac{\partial N}{\partial w} &= N_w = 2(n_{200}w + n_{110}u + n_{101}v) \\ \frac{\partial N}{\partial v} &= N_v = 2(n_{002}v + n_{101}w + n_{011}u) \\ \frac{\partial N}{\partial u} &= N_u = 2(n_{020}u + n_{110}w + n_{011}v). \end{aligned} \quad (6)$$

$$\dot{N} = \frac{d}{d(u \text{ or } v)} N = N_w \dot{w} + N_u \dot{u} + N_v \dot{v}. \quad (7)$$

$$dN = \frac{d}{d(u \text{ or } v)} \left(\frac{N}{\|N\|} \right) = \frac{\dot{N}\|N\| - N \frac{N \cdot \dot{N}}{\|N\|}}{\|N\|^2}. \quad (8)$$

Once the direction vector is scaled by distance and curvature, it is used to define a new set of barycentric coordinates. This is done by adding the scaled direction vector to a point being tracked across the surface of the triangle, and converting the result into barycentric coordinates. This result then becomes the tracked point for the next iteration.

Switching between base triangles is also done at this point. If the tracked point ever leaves the bounds of the current triangle, the focus is switched to the adjacent triangle which shares the crossed edge. The iterations continue as previously as though nothing occurred, only now, the computations use the new triangle. Additional switching needs to be monitored when there is the potential for contacting two nonadjacent triangles simultaneously.

4.5.3 Computing the New Tangent

Plane. After the new barycentric coordinates have been computed, the error needs to be evaluated to see if more iterations are needed. First, the new surface point and normal are computed. The error is defined as the dis-

tance between the new tangent and its proxy contact point. The ideal contact point is when the computed surface point and the contact point on the tangent plane are the same (thus error ≈ 0). If the distance between the proxy contact point and computed Bézier surface point is too large ($>1 \mu\text{m}$), the process is repeated again using the newly computed tangent plane. The convergence error of $1 \mu\text{m}$ was chosen to eliminate perceptible artifacts while still allowing reasonable convergence times. With properly tuned gains, the system takes, on average, two to three iterations to converge for the objects presented in Section 5.

5 Evaluation Experiments

5.1 Overview of Experiments

Two experiments were run using the 2D (see the Appendix) and 3D (see Section 4) shading algorithms, respectively. The first experiment evaluated several rendering conditions to obtain perceptual thresholds for rendering smooth objects. From the results of the first experiment and the 2D shading algorithm, the 3D algorithm was developed. The second test, involving the 3D algorithm, was used as a means of validating the 3D algorithm and providing further insight into the CLD device's capability to facilitate exploration and shape recognition within a 3D environment. All experiments were conducted with the approval of the University of Utah Institutional Review Board.

5.2 Smoothness Discrimination of 2D Polygonal Surfaces

5.2.1 Participants. Twelve right-handed individuals (three females) between the ages of 19 and 41 participated in this experiment. None of the participants had prior experience with PHANToMs or the CLD device.

5.2.2 Stimuli. The reference stimulus was a mathematically correct arc segment of a circle (see Figure 8), while the comparison stimulus was a polygonal approximation of the same arc segment. Only the top portion of the circle was haptically rendered. The rendered arc sec-

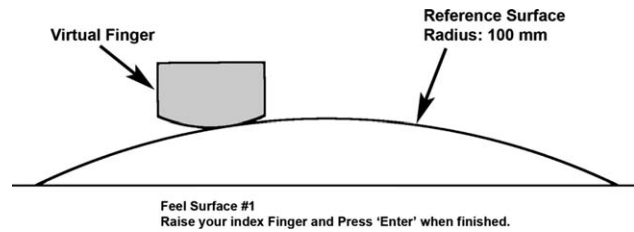


Figure 8. Screen capture of the smooth reference object used during training that preceded each test condition.

tion was 0.902 radians of a 100 mm radius circle, giving approximately 90 mm of travel space. Contact location on the virtual finger was calculated over a 16 mm arc length of the 20 mm radius finger model and linearly mapped to be displayed over 16 mm of travel along the length of the participant's finger.

5.2.3 Design. Four haptic rendering conditions (C1–C4) were evaluated in order to better understand the requirements for rendering smooth objects when using polygonal models. An adaptive procedure was utilized to assess when participants could no longer distinguish between the polygonal model and the smooth reference surface. These tests were conducted with kinesthetic feedback alone and with combined tactile and kinesthetic feedback. Force (kinesthetic) and tactile shading were also specifically investigated. Forces were rendered using a PHANToM Premium 1.5 while tactile feedback was rendered using the contact location display (CLD) device.

The first two conditions parallel the work by Morganbesser and Srinivasan (1996) and utilize solely kinesthetic force feedback. In these conditions, the contact roller of the contact location display was simply held at the middle of the thimble. Condition 1 (C1) utilized a set of polygons (line segments) to approximate a smooth surface, and did not use any haptic shading. This was done to establish a baseline for the number of segments required for a polygonal model to feel smooth.

Condition 2 (C2) was identical to Condition 1 (C1), but also included the addition of force shading, as described by Morganbesser and Srinivasan (1996). One slight difference from Morganbesser and Srinivasan was that we utilized a curved finger model as opposed to a

point contact virtual finger model. Completing this experimental condition extends the work described by Morganbesser and Srinivasan to a more complete state that can be more readily used by hapticians when constructing virtual models of smooth surfaces.

The remaining two conditions utilize the contact location display. Condition 3 (C3) has participants evaluate polygonal models with tactile and kinesthetic feedback (with no shading/smoothing) and the results can be compared to those of Condition 1 (C1) to examine the effect of added contact location feedback.

Condition 4 (C4) had participants utilize tactile and kinesthetic feedback to evaluate polygonal models with tactile shading, but without force shading. This condition was designed to evaluate the influence of tactile feedback and could be compared to all three other conditions. The reason that we did not run our experiment with both tactile and force shading was that we found that this condition resulted in a trivially short experiment during pilot testing (referred to as P1 in Section 5.2.6). That is, participants had difficulty distinguishing the shaded polygonal and perfectly smooth surfaces even when very few polygons were used, and our adaptive procedure would not be appropriate for evaluating this threshold condition. This was to be expected because at five line segments there was less than a 0.4% deviation in curvature between the shaded model and the actual smooth surface. Our pilot testing also indicated that adding force shading to force and contact location display (referred to as P2) provided no significant change in sensitivity and was not tested further.

5.2.4 Procedure. The experiment utilized a paired-comparison (two interval), forced-choice paradigm, with a 1-up, 2-down adaptive procedure (Levitt, 1971). On each trial, the participant was presented with two objects, the smooth reference object and the comparison object with a polygonal representation, in a random order. The participant's task was to indicate which of the two shapes was the smooth object. The number of line segments was decreased after one incorrect response (making the difference between the reference and comparison objects larger, and therefore the task easier) and increased after two consecutive correct responses



Figure 9. Experimental test setup (cover pulled back for clarity).

(making the task more difficult). The threshold obtained corresponds to the 70.7% confidence interval on the psychometric function (Levitt).

Each condition was conducted as follows. On each trial, the participant would first feel stimulus #1. Once he or she was finished exploring, the participant would then raise the index finger off the surface and press the Enter key to indicate readiness for stimulus #2. After feeling the second stimulus, the participant would again raise the index finger and press 1 or 2 and then Enter to indicate which of the two stimuli was the smooth object. Then, a new set of comparisons was presented. The order of the reference and comparison stimulus presentation was randomized.

The experiment continued until the participant had finished 11 reversals (a reversal occurred when the number of segments was increased after a decrease, or vice versa). A large step size was used for the first three reversals for a faster initial convergence. A reduced step size was used for the remaining eight reversals for better accuracy in determining the discrimination threshold. The step sizes for each condition were chosen during pilot testing and fixed for all participants in the study.

A Latin Squares reduction of the system was utilized to reduce the number of permutations for balancing the testing order in which the participants completed the four experimental conditions. The testing apparatus, as shown in Figure 9, was obscured by a cloth cover so that the user would not be able to see either the haptic or tactile device. Instructions were posted on the screen to remind the user where within each comparison they

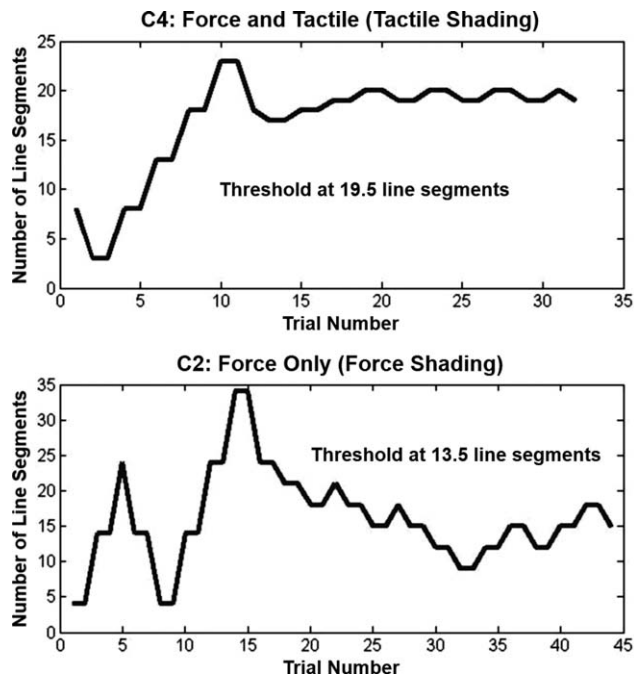


Figure 10. Two collected data plots showing (top) nearly ideal data from one participant and (bottom) less ideal data from the same participant who had difficulty with C2.

were, and how to proceed, but no other visual feedback was provided. White noise was played over headphones to block all auditory feedback, except for audio cues that were provided to indicate the transition between stimuli. Participants were given as much time as they desired to explore each stimulus, but were not permitted to go back to the first stimulus once they had proceeded to the second. It took an average of about 42 trials and 10 min to complete each condition per participant.

5.2.5 Data Analysis. Two representative data sets for one participant are shown in Figure 10. Note that this participant had some difficulty in C2 (force feedback with force shading). However, both of these plots still fall within the range of expected participant performance. In all cases, each participant managed to stabilize performance before completing the 11 reversals. Thresholds were computed as the average of the last six reversals.

5.2.6 Results. Table 1 shows the mean discrimination thresholds and the corresponding 95% confidence

intervals for the four experimental conditions. While our experiment evaluated the number of polygons needed for a polygonal surface to be indistinguishable from a reference smooth surface, the results are also reported in terms of the more general metric of the angle difference between adjacent polygonal surfaces. To best understand the practical implications of these data, it is useful to consider this example. If the angle difference between adjacent polygons in a model used exceeds the 95% confidence interval (e.g., $<0.37^\circ$ for C1) then 97.5% or more of people should sense the model as perfectly smooth. Note that the participants were concentrating on the smoothness, so if they were simultaneously engaged in other tasks, these thresholds would increase. Figure 11 plots these means and confidence intervals to visually highlight the significant differences among the four conditions.

The data collected from the 12 participants passed an omnibus ANOVA test, $F(44, 47) = 47.76$, $p < .001$. This implies independence between all four conditions and allows the use of Tukey's test to determine whether the results are significantly different. The data were subsequently analyzed for statistically significant differences using Tukey's test with $\alpha = .05$. The average number of line segments for each threshold was the highest for C3 (257.3), followed by that for C1 (104.1), and the lowest for C2 and C4 (16.3 and 15.6, respectively). It was found that C3 (force and tactile rendered) was significantly different from all other conditions. C1 (force only rendered) was also significantly different from all other conditions. The two shading conditions (C2 and C4) were not significantly different from each other.

As mentioned earlier, a more general and useful metric that can be taken from our results is the angle difference between adjacent polygons, as this can be applied to other generic polygon models. This measure corresponds to the way discontinuities between line segments connect. This concept is similar to that proposed by Morganbesser and Srinivasan (1996) with one important distinction: the tactile feedback is felt as short rolling bursts as the user crosses the vertexes, due not only to the instantaneous changes in force direction but also to changes in the geometric shape itself (e.g., angle differences between adjacent polygons). Table 1 shows the angle difference thresholds corresponding to the line

Table 1. Means and 95% Confidence Intervals for All Four Test Conditions, Showing the Number of Line Segments Needed for a Polygonal Surface to be Indistinguishable from the Smooth Reference Surface and the Corresponding Angle Difference Between Adjacent Line Segments in Degrees (in Parentheses)

	C1: Force only	C2: Force only with force shading	C3: Force and tactile	C4: Force and tactile with tactile shading
Mean	104.1 (0.5°)	16.3 (3.4°)	257.3 (0.2°)	15.6 (3.5°)
95% Confidence	±35.32 (+0.25°, -0.13°)	±1.99 (+0.44°, -0.35°)	±63.20 (+0.07°, -0.04°)	±3.85 (+1.09°, -0.66°)

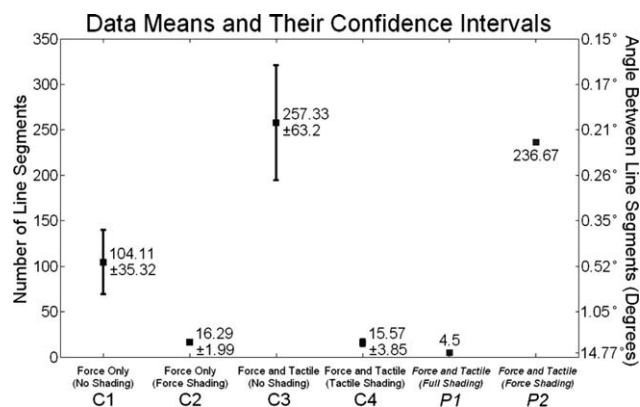


Figure 11. Mean and 95% confidence intervals for each test condition showing the number of line segments at which the polygonal model was indistinguishable from the smooth reference surface. The error bars are not linear when interpreting results based on the angle difference between segments.

segment thresholds in parentheses. The same angle differences are shown in Table 2, where test conditions are organized according to rendered and shaded variables. Two additional threshold values are shown from pilot testing (P1 and P2, collected from two participants) for comparison and discussion later.

5.2.7 Discussion. Our results are not directly comparable to the data of Morganbesser and Srinivasan (1996), as these researchers only tested to show improvements in perceived smoothness and explored coarse models using up to three polygons. However, it is interesting to compare C1 to prior work on discriminat-

Table 2. Estimated Mean Angle Difference, in Degrees, Between Adjacent Line Segments to Create a Curved Surface that Feels Smooth

	Rendered condition	
	Force only	Force and tactile
No shading	0.5° (C1)	0.2° (C3)
Force shading	3.4° (C2)	0.2° (P2)
Tactile shading	NA	3.5° (C4)
Force and tactile shading	NA	14.8° (P1)

ing the angle difference between sequentially applied force vectors. Barbagli, Salisbury, Ho, Spence, and Tan (2006) reported a discrimination threshold of 28.4° for sequentially applied force vectors, which is nearly two orders of magnitude larger than the thresholds we report for the instantaneous changes in force orientation experienced in C1 (0.5°). This is not surprising, though, as people have much greater sensitivity to changes presented in rapid succession (Gescheider, 1997). Our task also utilized active rather than passive sensing in making perceptual judgments, which is also expected to provide greater perceptual sensitivity (Klatzky & Lederman, 2003). Frisoli, Solazzi, Reiner, and Bergamasco (2011) performed an experiment involving both force and tactile feedback, which demonstrated that the addition of tactile feedback increased the user's ability to detect small angle differences between nearly parallel planes. Frisoli et al. reported a perception threshold ranging

from 0.7° with force and tactile feedback to 2.6° with force feedback only. Since our task involved detecting the edge formed from the two planes rather than detecting a change in force direction, we would expect our results to show lower perception thresholds (our results showing 0.2° to 0.5° thresholds under the same feedback conditions). Several trends can be observed from the data presented in Table 1 and Table 2. First of all, the addition of tactile feedback greatly increases one's sensitivity to edges and vertices in the system, as seen by pair-wise comparisons of the thresholds for C1 and C3 and those for C2 and P2 in Table 2. This increased sensitivity is undesirable when rendering smooth surfaces as it requires more line segments, causing an increase in computation time and a decrease in rendering performance. Fortunately, force and/or tactile shading can decrease one's sensitivity to edges and vertices, as seen by the significant difference found between the thresholds for C1 and C2 and those for C3 and C4. This significant difference shows that both the force shading algorithm, developed by Morganbesser and Srinivasan, and our 2D shading algorithm (presented in the Appendix), significantly reduce the needed number of line segments to make a polygonal object feel smooth. Further, it is not practical to provide tactile feedback for polygonal object models without our shading algorithm, as indicated by P2. Note that in C4, the 2D shading algorithm did not smooth forces. Therefore, the threshold of 3.5° can be further improved (in terms of decreased number of line segments) by employing force shading, as indicated by the threshold of 14.8° for P1 shown in Table 2.

Another interesting observation is that people appear to rely more on tactile than force information to judge the smoothness of a surface. Participants judged polygonal surfaces in C4 to be smoother based on shaded tactile feedback, even though normal force discontinuities still existed to the same degree as in C1. This indicates that the tactile sensations may carry more weight in the haptic perception of smoothness than the force irregularities. In fact, in the presence of unshaded tactile information (see C3 and P2 in Table 2), there appears to be no significant benefit from applying Morganbesser and Srinivasan's force shading algorithm.















In summary, the use of shading algorithms can lead to a significant reduction in the size of polygonal models by approximating smooth object surfaces without introducing noticeable artifacts. Very small angle differences between adjacent polygons ($0.2\text{--}0.5^\circ$) were required for rendering smooth objects when shading was not used. Thus, large numbers of polygons were needed for these models to feel smooth. The addition of force and/or tactile shading significantly reduced the required model size as can be seen in Figure 11 and Table 2. Either form of force (C2) or tactile (C4) shading allowed a relatively large angle difference between polygons ($\sim 3.5^\circ$, a factor of six over unshaded conditions), while our pilot tests (P1) showed that a greater angle difference between polygons ($\sim 15^\circ$, a factor of 30 over unshaded conditions) was possible if both force and tactile shading were simultaneously applied, thereby requiring a significantly smaller number of polygons to represent a given smooth haptic model. This can clearly have a huge impact on reducing the necessary size of a haptic model, without sacrificing the fidelity of the haptic interaction. Although our results were obtained with the contact location display (in C3 and C4), the angle difference thresholds are likely applicable to other types of tactile displays including those that render the tangent lines of a curved surface (Dostmohamed & Hayward, 2005; Frisoli et al., 2005).

5.3 Identification of 3D Object Shapes

5.3.1 Participants. Seventeen right-handed individuals and one left-handed individual (two females) between the ages of 18 and 38 participated in the experiments. The participants were divided into two groups: experienced and inexperienced. Experienced participants took part in the 2D experiment described in Section 5.2. Inexperienced participants had no prior experience with the CLD device but some had prior experience with other haptic devices. There were nine experienced and nine inexperienced participants.

5.3.2 Stimuli. Seven objects were selected for this experiment during pilot testing as being distinct, while providing opportunity for confusion between similarly

Table 3. Confusion Matrix Showing Percent Accuracy for All Participants*

Shape presented to participant	Shape identified by participant						
							
	83.3	4.5	0.3	1.7	8.3	0.3	1.4
	1.7	88.2	5.9	0	1.0	1.7	1.4
	2.1	3.1	81.6	2.1	0.7	4.9	5.6
	0.3	0.7	0	97.6	0.3	1.0	0
	20.1	0.3	0.3	3.8	70.5	1.0	3.8
	1.0	6.9	3.5	0	1.7	83.7	3.1
	2.4	4.2	3.8	0.7	10.1	10.4	68.4

*The diagonal has been highlighted in dark gray. Major confusion values have been highlighted in light gray.

shaped objects, depending on the rendering conditions. These seven primitives are the cone, cylinder, cube, sphere, tetrahedron, extruded octagon, and extruded triangle (see images in Table 3). Each object fit within a 40 mm radius cylinder and was 80 mm long, with the exception of the cube, which was 56.6 mm long. The orientation of these objects was fixed for all models in the experiment with the primary axis as horizontal and from the left to right. Participants were not informed of the model orientation to prevent exploration strategies involving finding a particular feature. The cone and tetrahedron models are asymmetric along this axis and could provide directional information. These models were rendered facing either direction (pointed to both left and right) during the experiment to eliminate the direction cue.

The 1-DOF gimbal on the CLD was modified from the first experiment (Section 5.2) to allow additional motion from side to side although only the tilt angle was monitored. The user's finger orientation was limited to pointing forward and tilting up and down.

5.3.3 Design. Virtual objects were rendered under four experimental conditions. The tests were conducted with either kinesthetic feedback alone or with combined tactile and kinesthetic feedback. Kinesthetic feedback was provided by a PHANToM device and tactile feedback was provided by a 1-DOF CLD device. Object models were rendered with or without haptic shading. The former case created smooth curved objects and rounded the edges of flat-sided objects such as cubes. Rounded corners with a radius of 1.5 mm were implemented as suggested at the end of Section 4.3 through the inclusion of extra triangles. See Doxon (2010) for further rendering details.

The addition of rounded edges was expected to allow the user to better maintain contact with the object's surface and thus improve object recognition. Loss of contact with objects was a problem that hampered participants' ability to identify simple object shapes as previously reported by Frisoli et al. (2005). Objects containing smooth curved surfaces (cone, cylinder, and sphere) were rendered as high count

polygonal representations when haptic shading was not used.

5.3.4 Procedure. A blocked design was utilized for this experiment. Each participant performed a total of eight runs across two sessions, containing four runs each. Each session was separated by at least a day. Within each run, the participant was presented with all seven objects as both shaded and unshaded models to identify. Each of the fourteen object models was presented once per run, and the order in which they were rendered was chosen randomly. Two runs (a block) were conducted back to back with the same stimulus set. Shapes containing directional information (cone and tetrahedron) were rendered facing either left or right and chosen such that across each session both directions were experienced under each rendering condition.

The first half (two runs) and second half (two runs) of each session differed in whether tactile feedback was rendered or not. Even numbered participants evaluated the first half of the experiment with tactile and kinesthetic feedback and the second half with only kinesthetic feedback, while odd numbered participants performed the opposite. When no tactile feedback was rendered, the CLD device was commanded to a position at the center of the thimble and remained in contact with the participant's finger to ensure a purely kinesthetic interaction.

In each trial, the participant explored the currently rendered object and identified it from the list of seven objects provided to them (see Table 3). The participant was instructed to press the number key corresponding to the identified shape, for example, 4 for a sphere. The response and timing data were recorded and the participant was guided back to the starting position by weak attractive forces and visual feedback of the finger position. Participants were required to remain at the starting position for 1 s before continuing. This helped the participant to begin each trial at the same relative location above each virtual object. At the end of the 1 s period, a ding sound was played and visual feedback disappeared to prompt the participant to begin exploring the next object to be identified. The experiment continued until all 14 objects in a run were identified. A short break was given between the second and third runs in a session

while the CLD device was adjusted for use in a different feedback condition, which involved the addition or elimination of tactile (CLD) feedback.

Before the test data were recorded for each feedback condition, the user was allowed to interact with an extruded hexagon for practice. Visual feedback showing the virtual object and the user's virtual finger on the LCD was provided to the user during the practice. However, no such visual cues were provided during the main experiment, except for the visual cues that guided the user to raise the finger back above the virtual objects after each response.

The same testing apparatus that was used during the 2D experiment (see Figure 9) was also used in the 3D object recognition experiment. A cloth cover was used to stop the user from being able to see either the haptic or tactile device. A list of the seven objects and their corresponding numbers was provided to the participants on a sheet of paper but no further instructions were posted on the screen. White noise was played over headphones to block all auditory cues except those provided by the program to indicate a transition between trials. Participants were given as much time as they desired to explore the objects, but were instructed to respond as quickly as they felt comfortable. Participants were not permitted to change their responses once given.

5.3.5 Data Analysis. Trials from all participants were pooled and organized into a stimulus-response confusion matrix with rows representing stimulus and columns representing responses. This matrix was further broken into two to show each combination of rendering conditions. These matrices were used to evaluate percent correct scores and pair-wise confusions as well as response time data. Only response times from correct answers were used to determine average response times.

5.3.6 Results *5.3.6.1 Accuracy.* Figure 12 shows the number of correct answers given by participants for each of the seven different objects. Objects were identified with a mean accuracy of 81.9% and *SD* of 10.0%. This matches the results found in Kirkpatrick and Douglas (2002) where a similar object identification task was performed (mean 84%, *SD* 12%). Jansson and Monaci

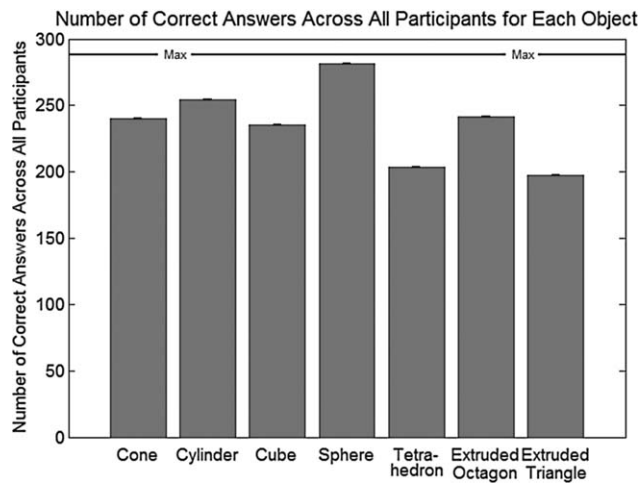


Figure 12. Total number of correct answers given by all participants for each of the seven objects.

(2006) found an accuracy of around 70% when exploring real objects with a plastic shell placed over the fingertip. This relatively high percent-correct score indicates that a performance ceiling may have been reached, making it difficult to observe any performance improvement in accuracy due to the additional tactile (CLD) cues. Of the seven shapes, the extruded triangle was the most difficult to identify at a 68.4% accuracy and the sphere was the easiest at a 97.6% accuracy. These values can also be computed from the diagonal cells of Table 3.

Table 3 shows the stimulus-response confusion matrix in percent-correct scores pooled from all participants. The rows represent the stimulus shapes presented to the participant while the columns represent the responses. The diagonal cells containing correct answers have been highlighted. Significant off diagonal terms have been bolded and shaded. Compared to a chance performance level of 14.3% (1/7) correct, the overall accuracy was relatively high, indicating that the participants were able to disambiguate the seven test stimuli reasonably well. The off-diagonal cells in Table 3 are asymmetric, which implies that participants perceived some objects as others but not vice versa. The most predominant confusion was identifying the tetrahedron as a cone (20.1% of the total trials) and to a lesser extent the cone as a tetrahedron (8.3% of the total trials).

Weaker (<10%) but still predominant confusion was also observed. Participants confused the extruded octagon with the cylinder (6.9% of the total trials) more often than vice versa (1.7% of the total trials). The extruded triangle was confused for the tetrahedron (10.1% of the total trials), which contains a similar shape and orientation. While all the listed confusions so far are between elements with similar geometry, the confusion between the extruded triangle and the extruded octagon (10.4% of the total trials) was unexpected. One reason for this confusion may have been that the extruded triangle's faces are nearly vertical, which makes them more difficult to interact with. Participants may have been identifying the shape as an extruded octagon by the orientation of the faces alone, rather than fully comprehending the overall shape of the model.

To the least extent there were small confusions involving the cone identified as a cylinder (4.5% of the total trials), the cylinder identified as a cube (5.9% of the total trials), and the cube identified as an extruded octagon (4.9% of the total trials) and extruded triangle (5.6% of the total trials). These confusion elements constitute less than 6% of the total number of trials for each object.

5.3.6.2 Effect of Haptic Shading. The confusion matrix shown in Table 3 was split into two matrices according to whether the object's edges were rounded. It was found that shading had no effect on the confusion of the tetrahedron with other objects, between the extruded triangle and the extruded octagon, or the confusion of the cylinder as the cube.

However, the extruded octagon was predominantly confused with the cylinder when its edges were rounded (shaded), whereas the following confusions mainly occurred with unshaded objects: the extruded triangle as the tetrahedron, and the cube as either the extruded octagon or the extruded triangle. Overall, there was not a significant difference found in accuracies between objects with and without rounded edges, $t(502) = 1.53$, $p = .1277$.

5.3.6.3 Effect of CLD. The confusion matrix shown in Table 3 was subdivided into two matrices to examine the effect of additional contact location feed-

back on object recognition. The percent-correct scores were 82.5% and 81.3% for the kinesthetic alone and combined kinesthetic and tactile feedback cases, respectively. Neither the identification accuracy nor response time was significantly different. Jansson and Ivas (2001) indicated that the potential usefulness of a device may be underestimated when inexperienced users are evaluated. The potential ceiling effect coupled with the fact that the majority of users were not explicitly trained on the device could explain the lack of significant difference. This was somewhat in contrast with our findings in the first experiment reported in Section 5.2.

5.3.6.4 Effect of User Experience. The confusion matrix in Table 3 was also divided into two matrices for the experienced and inexperienced participants. The overall percent-correct scores for the experienced and inexperienced participants were 87.5% and 76.3%, respectively. Experienced participants were significantly more accurate than inexperienced participants, $t(250) = -4.01$, $p < .0001$. While the weaker confusions were not present for the experienced users, both groups had the same level of difficulty identifying the extruded triangle.

5.3.6.5 Response Time. Two types of response times were collected within each trial. The first of these began counting as soon as the object was touched and haptic forces were rendered. The second gathered response time counted only during the times when the user was in contact with the surface of the object. Both response times stopped counting when a response was given. This response time data provides additional measures of the difficulty of the object identification task. Figure 13 shows the mean times between the start of a trial and when a participant responded for all seven objects. The average response time varied from 8.6 s (sphere) to 18.7 s (extruded triangle), with the sphere taking only about half the amount of time to identify as any of the other six objects, $t(1649) = -10.10$, $p < .0001$, and being significantly more accurately identified, $t(250) = -4.62$, $p < .0001$. This was as expected because of the sphere's unique geometric profile among the seven objects. Kirkpatrick's 2002 object identification task provided similar identification times of 22.4 s.

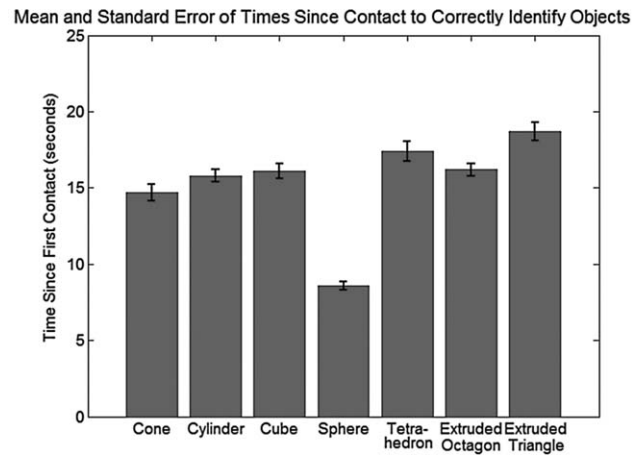


Figure 13. Time from initial contact to response for each of the seven objects.

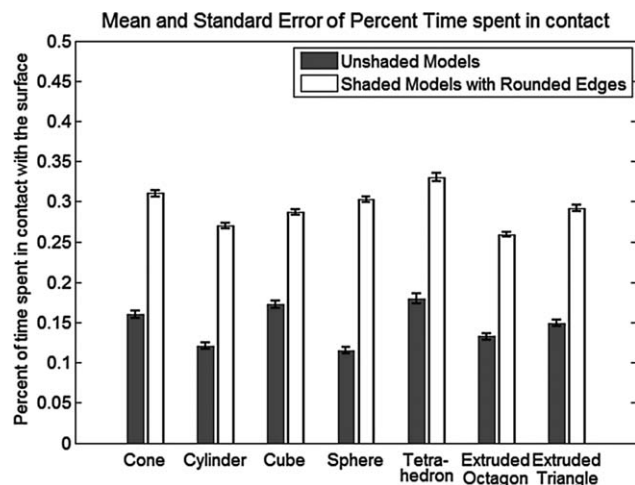


Figure 14. Effect of shading on the percent time spent in contact with objects.

5.3.6.6 Effects of Haptic Shading. The effect of shading on object recognition time can be seen in Figure 14 where the percent of time in contact with the object under the shaded and unshaded conditions is shown. It can be seen that rounded edges on objects allowed participants to stay in contact with the object's surface for a larger portion of the total object-exploration time for each of the seven stimulus objects, $t(1649) = 37.14$, $p < .0001$. However, as mentioned earlier, the longer contact time for shaded objects did not result in a significantly higher object-recognition accuracy level.

5.3.6.7 Effect of User Experience. Response time data for experienced and inexperienced users was compared. Experienced users were found to be universally faster at identifying the objects $t(1649) = -5.92$, $p < .0001$. All objects showed a significant difference in identification time except for the extruded octagon and tetrahedron. Experience made the largest time difference on the extruded triangle.

5.3.7 Discussion. The results of the 3D object recognition experiment showed that the participants were able to identify seven common geometric shapes with an accuracy of above 80% correct with force and contact location information. With this relatively high recognition rate, we might have hit a ceiling effect that made it difficult for the participants to demonstrate any additional benefit of 3D shading of object edges or contact location information. More detailed analysis of the confusion matrices showed that while shading reduced confusion between objects for some shapes (e.g., misrecognition of cubes as extruded triangles or extruded octagons), it did not significantly affect the recognition accuracy for tetrahedrons. Moreover, shading appeared to have contributed to increased confusion of extruded octagons as cylinders. Some of these results are as expected. For example, users likely had a difficult time following the contours of the cube while it was unshaded. The addition of rounded edges eliminated this problem and therefore made cubes more distinguishable from extruded triangles or extruded octagons. Other results, such as that for the tetrahedron, appear to suggest that tetrahedrons are generally difficult to recognize with the experimental setup used in the present study.

Our results showing that the addition of rounded edges significantly increased the percentage of time spent in contact with virtual objects are consistent with those of other studies. For example, Frisoli et al. (2005) previously reported that loss of contact with objects hampered their subjects' ability to identify simple object shapes.

Users with prior experience with the CLD device identified objects faster and with higher accuracy than those without. This finding indicates that, like other

haptic devices, the CLD device required some practice before it can be used to its fullest potential.

Independently, participants seemed to develop a common exploration strategy. This strategy involves first moving left and right to determine whether there are sides on the object. This was done using only kinesthetic information due to finger orientation and the CLD device characteristics. This immediately determines which of three groups the object falls into: (1) the sphere, (2) the cone and tetrahedron, and (3) the cylinder, cube, extruded octagon, and extruded triangle. Participants then returned to the center of the object and explored forward and backward to identify the object from within the subgroup. This exploration strategy explains the faster speed and better accuracy in identifying the sphere as it is unique in the left-right direction. The strategy also indicates why potential confusion may have occurred on the extruded triangle and tetrahedron, which both contain only an edge along the top.

It was expected that the use of the CLD device would decrease confusion among the objects due to the additional tactile cues. The results show that while there is no statistical difference between the number of correct answers given with and without tactile feedback, the majority of the off-diagonal confusion cells identified earlier in Table 3 are more uniformly distributed when tactile feedback is presented, indicating less overall confusion. While the tactile cues might have assisted the participants in object recognition, users' interactions with the CLD device suggest that further mechanical revisions are required before the CLD can provide more effective haptic interactions in 3D environments. This was especially noticeable when using the CLD to contact the front or bottom faces of objects. In this situation, the dynamics of the device bend the spring steel drive wires away from the user's finger and conflicts with the intended haptic interaction. Therefore, whatever benefits the CLD device provided might have been degraded by the limitations in its mechanical design.

6 Conclusions and Future Work

We have presented haptic shading algorithms that make it possible to fully utilize the CLD device with

polygonal object models. These algorithms can also be used with other haptic systems with combined tactile and kinesthetic feedback. Haptic shading algorithms for both 2D and 3D environments were developed. Both algorithms create perceptibly smooth haptic interactions allowing a significant reduction in the size of complex models. These algorithms can serve as a replacement to Morganbesser and Srinivasan's (1996) force-shading algorithm for a range of haptic devices. Each haptic shading algorithm was evaluated experimentally.

The experimental results are intended to be used as a guide to utilizing haptic shading to its fullest extent. The rendering thresholds provided through the first experiment state the level of detail haptic models needed in order to feel smooth when rendered with general kinesthetic and/or tactile rendering systems. The first experiment, utilizing the 2D algorithm, evaluated the perception thresholds for angle difference between adjacent polygons under four cases: unshaded force rendering, shaded force rendering, unshaded force and tactile rendering, and shaded tactile with unshaded force. The addition of tactile feedback through the CLD device significantly increased the ability of users to detect an edge from 0.5° to 0.2° angle difference between adjacent polygons. The inclusion of shading in both tested conditions substantially decreased the perception threshold and allowed the angle between adjacent polygons to increase to $\sim 3.5^\circ$. The full shading algorithm was found to reduce this further, allowing up to $\sim 15^\circ$ angle difference between adjacent polygons before model discontinuities became noticeable.

A second experiment, utilizing the 3D algorithm, evaluated the CLD device's capability to facilitate dexterous exploration and shape recognition. This experiment demonstrated the efficiency of our 3D algorithm, but points out design flaws in the current CLD device.

Our experiments indicate that the CLD device should be revised before conducting further tests in 3D environments. Such a redesign will permit research into grasping and manipulation. The next revision of the device may need to apply kinesthetic feedback through the thimble rather than through the contact element (roller) of the CLD device. After redesigning the device to make it more effective within 3D environments, there may be a

more noticeable improvement in user capability to identify objects rendered with contact location feedback.

Acknowledgments

This work was supported, in part, by the National Science Foundation under awards IIS-0904456 and IIS-0904423. The authors thank Jaeyoung Park for suggesting a more concise method of expressing the angular fraction used within the 2D algorithm (see Appendix).

References

- Barbagli, F., Salisbury, K., Ho, C., Spence, C., & Tan, H. Z. (2006). Haptic discrimination of force direction and the influence of visual information. *ACM Transactions on Applied Perception*, 3(2), 125–135.
- Cohen E., Riesenfeld, R., & Elber, G. (2001). *Geometric modeling with splines: An introduction* (chaps. 5–11). Natick, MA: AK Peters.
- Daniels, J., Silva, C. T., Shepherd, J., & Cohen, E. (2008). Quadrilateral mesh simplification. *ACM Transactions on Graphics*, 27(5), 148.
- Dostmohamed, H., & Hayward, V. (2005). Trajectory of contact region on the fingerpad gives the illusion of haptic shape. *Experimental Brain Research*, 164(3), 387–394.
- Doxon, A. (2010). *Force and contact location shading methods for use within two and three dimensional environments*. Master's Thesis. UMI Order Number: AAT1478013, The University of Utah.
- Frisoli, A., Bergamasco, M., Wu, S., & Ruffaldi, E. (2005). Evaluation of multipoint contact interfaces in haptic perception of shapes. Multi-point interaction with real and virtual objects. *Springer Tracts in Advanced Robotics*, 18, 177–188.
- Frisoli, A., Solazzi, M., Reiner, M., & Bergamasco, M. (2011). The contribution of cutaneous and kinesthetic sensory modalities in haptic perception of orientation. *Brain Research Bulletin*, 85, 260–266.
- Frisoli, A., Solazzi, M., Salsedo, F., & Bergamasco, M. (2008). A fingertip haptic display for improving curvature discrimination. *Presence: Teleoperators and Virtual Environments*, 17(6), 550–561.
- Fritsch, M., Ernst, M., & Buss, M. (2006). Integration of kinesthetic and tactile display—A modular design concept. *2006 EuroHaptics Conference*, 607–612.

- Gescheider, G. A. (1997). *Psychophysics: The fundamentals*. (3rd ed.). Mahwah, NJ: Erlbaum.
- Jansson, G., & Monaci, L. (2006). Identification of real objects under conditions similar to those in haptic displays: Providing spatially distributed information at the contact areas is more important than increasing the number of areas. *Virtual Reality*, 9(4), 243–249.
- Jansson, G., & Ivas, A. (2001). Can the efficiency of a haptic display be increased by short-time practice in exploration? *Proceedings of Haptic Human-Computer Interaction Workshop*, 22–27.
- Johnson, D., & Cohen, E. (1999). Bound coherence for minimum distance computations. *Proceedings of IEEE International Conference on Robotics and Automation*, 1843–1848.
- Kirkpatrick, A., & Douglas, S. (2002). Application based evaluation of haptic interfaces. *Proceedings of the Tenth Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 32–39.
- Klatzky, R., & Lederman, S. (2003). Touch. In A. F. Healy and R. W. Proctor (Eds.), *Handbook of Psychology, Vol. 4: Experimental Psychology* (chap. 6, pp. 147–176). New York: Wiley.
- Kuchenbecker, K., Provancher, W., Niemeyer, G., & Cutkosky, M. (2004). Haptic display of contact location. *Proceedings of the IEEE Haptics Symposium*, 40–47.
- Levitt, H. (1971). Transformed up-down methods in psychoacoustics. *Journal of the Acoustical Society of America*, 49, 467–477.
- McNeely, W., Puterbaugh, K., & Troy, J. (1999). Six degree-of-freedom haptic rendering using voxel sampling. *Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1999)*, 401–408.
- Morganbesser, H., & Srinivasan, M. (1996). Force shading for shape perception in haptic virtual environments. *Proceedings of the 5th Annual Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 58.
- Phong, B. (1973). *Illumination for computer-generated images*. Doctoral Thesis. UMI Order Number: AAI7402100, The University of Utah.
- Provancher, W., Cutkosky, M., Kuchenbecker, K., & Niemeyer, G. (2005). Contact location display for haptic perception of curvature and object motion. *International Journal of Robotics Research*, 24(9), 691–702.
- Provancher, W., & Sylvester, N. (2009). Fingerpad skin stretch increases the perception of virtual friction. *IEEE Transactions on Haptics*, 2(4), 212–223.
- Ruspini, D., & Khatib, O. (2001). Haptic display for human interaction with virtual dynamic environments. *Journal of Robotic Systems*, 18(12), 769–783.
- Ruspini, D., Kolarov, K., & Khatib, O. (1997). The haptic display of complex graphical environments. *Computer Graphics and Interactive Techniques (SIGGRAPH 1997)*, 345–352.
- Salada, M., Colgate, J., Vishton, P., & Frankel, E. (2005). An experiment on tracking surface features with the sensation of slip. *WHC 2005, First Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 132–137.
- Seong, J. K., Johnson, D., Elber, G., & Cohen, E. (2010). Critical point analysis using domain lifting for fast geometry queries. *Journal of Computer-Aided Design*, 42(7), 613–624.
- Thompson, T., II, & Cohen, E. (1999). *Direct haptic rendering of complex trimmed NURBS models*. Paper presented at Haptic Interfaces for Virtual Environment and Teleoperator Systems.
- Vlachos, A., Peters, J., Boyd, C., & Mitchell, J. (2001). Curved PN triangles. *Symposium on Interactive 3D Graphics*, 159–166.
- Webster, R., Murphy, T., Verner, L., & Okamura, A. (2005). A novel two-dimensional tactile slip display: Design, kinematics and perceptual experiments. *ACM Transactions on Applied Perception*, 2(2), 150–165.

Appendix: 2D Haptic Shading Algorithm

A.1 Overview of the 2D Haptic Shading Algorithm

The 2D haptic shading algorithm creates a smooth haptic interaction given a 2D polygonal model. This is done by calculating a series of quadratic Bézier curves to create a new smooth curve based on the shape of the original polygonal model, which is then used to compute contact positions and rendered forces. This makes the underlying facets of the model imperceptible and allows a substantial reduction in model complexity while still retaining proper contours.

Rather than interacting with the Bézier curve directly, this approach computes a dynamically updated tangent line at the point of contact. To guarantee that the resulting smooth curved surface is continuous, all defined ver-

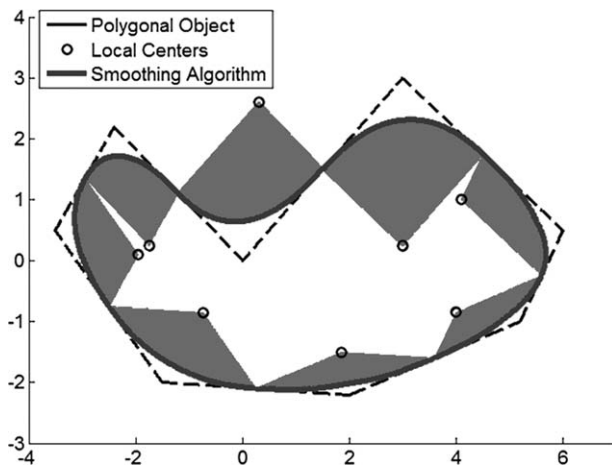


Figure 15. The original polygonal model (dashed black) and the smooth interaction model (thick curve). Separate Bézier patches are defined across each region denoted by the gray regions.

tices must be connected in a single polygon. Multiplicity, or multiple points defined at the same coordinates, can be used to generate sharp corners on the rendered smooth surface when desired.

While the algorithm was developed to provide both smooth tactile and kinesthetic feedback, it can be used as a substitute for the methods presented by Morganbesser and Srinivasan (1996) for force shading.

An example rendered smoothed surface for an arbitrary polygonal model is shown in Figure 15. The dashed black lines represent the original polygonal model and the thick curve represents the shape of the resulting curved surface. The gray shaded regions show the extent of each Bézier patch as well as a local parameterization used in the algorithm. The overall shape is built from these patches.

A.2 Bézier Curves

The 2D haptic shading algorithm utilizes a quadratic Bézier curve for each of its patches. Quadratic Bézier curves are defined by a control polygon containing three ordered points and have two valuable properties that help define the generated control polygon. First, the end points of the resulting Bézier curve are the end points of the control polygon (Cohen et al., 2001). Second, the

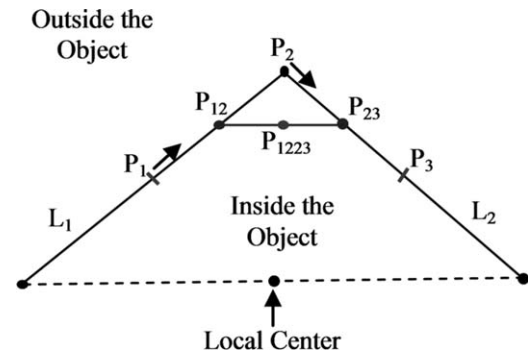


Figure 16. Basic labeling scheme used in our 2D shading algorithm.

quadratic Bézier curve is tangent to its control polygon at the end points (Cohen et al., 2001). These properties are used to guarantee that the resulting surface is smooth and contiguous.

The de Casteljau algorithm is an elegant constructive algorithm that computes a point and tangent on the Bézier curve based on a single parameter value, t (Cohen et al., 2001). Varying the parameter value from 0 to 1 traces out the Bézier curve. The de Casteljau algorithm allows us to directly compute the tangent line for any given value of t . Equation A1 defines the two points that make up this tangent line.

The labels used in these equations correlate to those shown in Figure 16. The point subscripts help to denote the location of the point. The two line segments that are adjacent to the vertex of interest are labeled L_1 and L_2 . The arrows denote the direction that the points P_{12} and P_{23} will travel for increasing values of t . The local center is an integral part of the radial parameterization used by the algorithm.

$$\begin{aligned} P_{12} &= P_1(1-t) + P_2t \\ P_{23} &= P_2(1-t) + P_3t. \end{aligned} \quad (\text{A1})$$

A.3 Preparing the Model

A.3.1 Defining the Control Polygon. In order to retain tangent continuity over patch boundaries, our algorithm forms a separate Bézier patch for each vertex on the original model. The control polygon is defined as

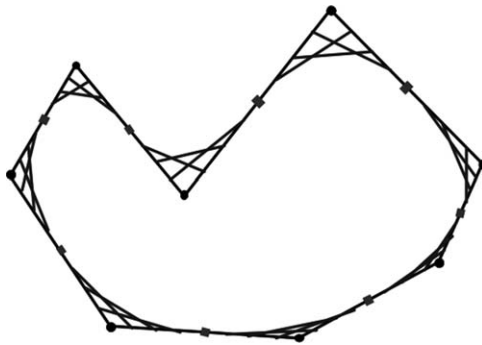


Figure 17. An arbitrary polygonal shape. Three tangent line segments are shown for each Bézier patch at $t = 0.25, 0.50,$ and 0.75 . Only one tangent will be in existence at a single instant in time.

the vertex and the midpoints of each line segment connected to it. Figure 17 shows three tangent line segments at $t = 0.25, 0.50,$ and 0.75 for each Bézier patch. The adjacent midpoints used are shown as tick marks.

Additionally, a single local center needs to be defined for each Bézier patch. This local center will be used to compute the new parameter value t in the algorithm. The local center cannot be located on $L_1, L_2,$ or the resulting curve. While the local center may be placed almost anywhere, ideally it should be placed at the center of curvature of L_1 and L_2 . The center of curvature can be found by computing the intersection of lines perpendicular to L_1 and L_2 placed at their respective midpoints. Placing the local center at the center of curvature of the polygonal lines ensures the highest numerical precision. Another convenient location for the local center is at the midpoint of the ends of L_1 and L_2 opposite the shared vertex, as used in Figure 17.

A.4 Implementing the 2D Haptic Shading Algorithm

The next few sections cover each step of the 2D haptic shading algorithm in detail.

A.4.1 Computing the Current Proxy Contact Location. To begin each iteration, the finger is projected into contact with the current tangent line. When moving, this contact position represents a small differential distance along the tangent line and thus is a reasona-

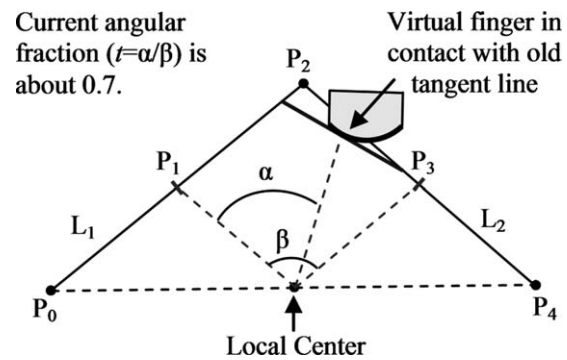


Figure 18. Computing the angular fraction based on the active line segments.

ble first approximation for determining the user's current position on the surface. No forces need to be computed or applied during this step.

A.4.2 Computing the New Parameter Value t .

From the new contact location, a parameter value t can be computed. Finding the parameter value of the Bézier curve that corresponds to the ideal contact point on the quadratic curve is difficult and slow. Instead, the parameter value is approximated through a radial parameterization, which slightly alters the shape of the resulting surface.

The first step in approximating the new parameter value t is to determine which Bézier patch to use. That is, determine the current L_1 and L_2 lines. These lines are likely the same ones as those from the previous iteration. There are two conditions that will cause new lines to be selected. The first of these conditions is when multiple contact points exist on nonadjacent line segments. The second condition occurs frequently just as the user passes over the midpoint of L_1 or L_2 . At this point, a new vertex is now closer to the contact point, and its corresponding line segments become the new L_1 and L_2 . The corresponding local center for the new Bézier patch is used.

Once L_1 and L_2 have been identified, all that is left is to compute the corresponding parameter value. This is done directly by computing the angular fraction ($t = \alpha/\beta$) between the current contact point and the start of the curve with respect to the local center. In Figure 18, the angular fraction is approximately 0.7.

Equation A2 shows how to calculate the parameter value t . Note that the angular fraction found when the proxy contact point lies directly on P_1 and P_3 will be either 1 or 0. This guarantees the resulting curve will end at P_1 and P_3 as well as being parallel to L_1 and L_2 at its ends. This allows the resulting curve to join adjacent Bézier curve patches with G^1 continuity.

$$t = \frac{\alpha}{\beta} = \frac{\theta_{P_1} - \theta_{\text{contact}}}{\theta_{P_1} - \theta_{P_3}}. \quad (\text{A2})$$

A.4.3 Computing the New Tangent Line. The last step is to compute the new tangent line segment by inputting the computed parametric value t into Equation A1. This tangent line is then used to compute haptic feedback. As the user reaches the midpoint of L_1 or L_2 , he or she also reaches the end point of the tangent line segment. Thus, the tangent line segment should always be extended to eliminate any artifacts that could be felt at this boundary.