FORCE AND CONTACT LOCATION SHADING METHODS FOR USE WITHIN TWO- AND THREE-DIMENSIONAL POLYGONAL ENVIRONMENTS

by

Andrew John Doxon

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Electrical and Computer Engineering

The University of Utah

August 2010

STATEMENT OF THESIS APPROVAL

[redacted]

has been approved by the following supervisory committee members:

William R. Provancher, Chair — 6/15/2009
Date Approved

Cynthia Furse, Member — 6/15/2009
Date Approved

Marc Bodson, Member — 6/15/2009
Date Approved

[redacted]

the Department of [redacted]

and by Charles A. Wight, Dean of The raduate School.

ABSTRACT

Current state-of-the-art haptic interfaces only provide kinesthetic (force) feedback. Providing tactile feedback in concert with kinesthetic information can dramatically improve one'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), that is capable of rendering (displaying) the center of contact to a user. The chief goal of this work is to develop algorithms that allow the CLD to be used with polygonal geometric models, which are commonly used to model haptic virtual environments. However, the use of polygonal models with the contact location display introduces an issue – the addition of tactile feedback enhances the saliency of the vertices and edges of polygonally represented (smooth) objects. That is, one can be distracted by the tactile sensations rendered when passing over the edges of the polygonal model. Two haptic "shading" algorithms were developed to address this issue and each provides smooth tactile and kinesthetic feedback.

The first shading algorithm was developed for a two-dimensional environment for a proof of concept and because the current CLD device is only capable of rendering contact positions lengthwise along the finger. Performance thresholds that specify the required number of polygons to render a smooth curve were evaluated in an experiment. It was found that the addition of contact location feedback significantly increases user sensitivity to small facets and features and the developed algorithm proved to create smoother interactions than prior shading techniques.

The second shading algorithm was developed to expand the first algorithm to three dimensions. An object recognition task was performed as both a demonstration of the 3D haptic shading algorithm and to provide further insight into the current CLD's capability to facilitate exploration and shape recognition. The study found the shading algorithm increased the amount of time users were able to stay in contact with the objects; however, these experiments also suggest that the current CLD device needs further mechanical revisions before it can provide effective haptic interactions in 3D environments.

Both of the developed shading algorithms create smooth haptic surfaces with accurate geometric representations and successfully extend the use of the contact location display to polygonal models. The presented results provide guidelines for future work and revision of the CLD device.

TABLE OF CONTENTS

APPENDICES

ACKNOWLEDGMENTS

I would like to thank all the people I have learned from throughout my life. I would not be the person I am today without their support and encouragement. I continue changing and developing on a daily basis and it is these people around me who make me who I am. I want to thank all of you from the bottom of my heart.

I especially want to thank my parents who have now dedicated a full 24 years of their lives to helping me become who I am. They taught me the importance of hard work, determination, and commitment. They support me when I have difficult decisions to make while giving me the freedom to make my own choices. I am very lucky to have them. I would especially like to thank my father who has been with me every step of my journey through graduate school asking me important questions and forcing me to explain my research to him. I would like to thank my younger brother who has chosen a life outside engineering that lets me see the world from an alternative viewpoint.

I would like to thank the many teachers I have made special bonds with over the course of my education. They enabled me to continue feeding my natural curiosity about the world and everything in it. I especially want to thank Dr. William Provancher, my advisor, for admitting me into his research group and giving me opportunities to learn and grow. He has taught me so many things in the short time I have known him, and for that, I am beyond grateful. I would also like to thank David Johnson for working with me on developing the techniques presented in this thesis and vetting my ideas.

I want to thank my lab mates Brian, Mark, and Scott for the aid they have given me. They are always willing to provide critique of my ideas and are wonderful to be around. We each take our own path but we all get there together.

Finally, I would like to thank the National Science Foundation for their grant money under award IIS-0904456 which made my stay at the University of Utah and this research possible.

CHAPTER 1

INTRODUCTION

Human computer interfaces which involve the sense of touch, or haptic interfaces, are becoming more and more used 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 et al. [7]. If implemented well, providing tactile feedback in combination with kinesthetic information should dramatically improve one's ability to dexterously interact and explore virtual environments. This thesis develops techniques to advance this goal.

One such device that provides both tactile and kinesthetic feedback is the contact location display (CLD) developed by Provancher et al. [26]. In addition to forces, this device displays the contact location between a virtual finger and surface on the user's finger. Figure 1.1 shows the concept for a contact location display. Previously, the CLD device was utilized only with specialized two-dimensional models. Use of polygonal geometric models, as is common with both haptics and computer graphics [27], with the CLD device would significantly expand the device's usefulness by allowing more diverse environments and future research opportunities. However, when interacting with polygonal approximations to smooth surfaces, the CLD transmits the surface

Figure 1.1 Concept for contact location feedback. The (left) two-dimensional or (right) one-dimensional center of contact is represented with a single tactile element.

discontinuities to the user. This gives the impression that the surface is meant to be rough rather than smooth and is distracting to the user even when high count polygonal models are used. The use of shading algorithms not only reduces the effects of the surface discontinuities but can 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 [23], smoothes the kinesthetic feel of faceted models by altering the surface normals as though the person were contacting a smooth surface, similar to Phong shading in graphics (see section 2.3) [25]. Modifying only the force direction presented still provides discontinuities in the form of proprioceptive (position) cues. Humans in general find it difficult to detect these proprioceptive cues so the smooth force interactions override these signals and cause the perception of a smooth object. However, contact location displays provide a distinct feedback modality, contact location, that is both sensitive to polygonal edges and not smoothed by Morganbesser and Srinivasan's force shading. The state-of-the-art is incapable of eliminating these discontinuities in the tactile feedback of the CLD device.

This thesis presents two haptic shading algorithms to provide smooth tactile and kinesthetic feedback for use within two- and three-dimensional polygonal environments. In this case, providing smooth tactile interactions is done by smoothing out the motions of the computed contact location. The first shading algorithm was developed for use in a 2D environment due to limitations of the current CLD device. This device is capable of motion only along the length of the finger and thus only capable of representing 2D contact locations. The second shading algorithm was developed to expand the 2D haptic shading algorithm to 3D polygonal environments. Both algorithms were implemented by creating a control polygon/mesh from the base polygonal model and using this to generate a smooth curved surface to interact with. Only the tangent plane at the point of contact was rendered to decrease computation time as finding the contact location between two curved surfaces requires robust numerical methods that may run too slowly for haptic applications [15].

1.1 Thesis Overview

There are seven chapters to this thesis. The following is a brief summary of each chapter and its contribution to this thesis.

Chapter 1 describes the motivation for this research and gives a brief overview of what that research entails. The two developed algorithms are each discussed in subsequent chapters followed by their related experiments.

Chapter 2 provides the current state-of-the-art in the field and prior work as well as a detailed description of the current contact location display device used in this thesis.

Chapter 3 presents detailed information about the developed 2D haptic shading algorithm. Bézier curves and the de Casteljau algorithm that are central to this shading algorithm are explained. The developed algorithm is then evaluated to determine its capabilities.

Chapter 4 contains information about the first experiment which determines discrimination thresholds for polygonal smooth surfaces. Four haptic rendering conditions were evaluated to determine the angle difference between adjacent polygons needed to render smooth objects when using polygonal models under these conditions. Detailed methods and procedures are given. The addition of tactile feedback is shown to greatly improve sensitivity to edges. The 2D haptic shading algorithm is shown to provide substantial benefit.

Chapter 5 provides detailed information about the developed 3D haptic shading algorithm. PN triangles are explained as well as alternative methods for computing curvature.

Chapter 6 details the second experiment which investigated object shape recognition to provide further insight into the CLD device's capability to facilitate exploration and shape recognition. Again, four haptic rendering conditions were evaluated. Detailed methods and procedures are given. Rounding edges on models allows users to better follow surface contours. The current CLD device requires further revisions before it is effective in 3D environments.

Chapter 7 presents a summary of the thesis and discusses the conclusions reached through this research. Future work and applications of this research are also discussed.

CHAPTER 2

BACKGROUND

This chapter provides a brief background concerning literature that is most relevant to this thesis research. Laycock and Day provide a brief overview of various haptic rendering techniques in [19].

2.1 Algorithms for Haptic Rendering

Haptics has its own distinct set of requirements compared to computer graphics and simulation. Generally, haptic force updates must occur around 1000 Hz in order to maintain stability while rendering “hard” contacts and to make controller updates imperceptible to humans [1]. Maintaining this high rate requires efficient programs and fast computers. Luckily, haptic computations are generally proximal to the user so resources can be focused locally around the contact locations.

The basic model for haptic rendering is to generate forces based on a virtual spring model that uses the amount of penetration between the virtual finger model and the environment to calculate the deflection of a virtual spring. In this case, the further the virtual finger model is pushed into the environment the harder the haptic device resists. The penetration distance between the virtual finger model and environment is generally estimated. Computing the penetration distance between two models is still an ongoing research topic.

2.1.1 Point-Model Haptic Rendering

The earliest haptic rendering methods approximated the user's finger as a point. This reduces the computational load to finding the distance between a point and model. The very earliest methods utilized vector fields in the environment to directly generate forces [11]. This approach proved to have several problems. Thin objects could be easily pushed through, and computing vector fields for all but simple objects is very difficult. Another early approach decoupled the depth computation from the model geometry by computing an intermediate plane [1]. This plane was based on the boundary surface of the model and updated at a slower rate than the force computations. Forces were generated using a spring model whose virtual spring is displaced by the distance from this intermediate plane.

The potential of this second technique was fully realized in the “god-object” and “proxy” approaches [30, 28, 27]. These approaches constrain a proxy (copy) of the virtual finger point to remain on the surface of the environment model (i.e., virtual surface). This proxy is pulled along the virtual object's surface toward the virtual finger point as the finger moves, becoming trapped within local surface minima. Normal forces are rendered in the direction from the virtual finger point toward its proxy. Penetration depth is the distance between the virtual finger model and its proxy.

Concurrent work on haptic rendering of smooth sculptured surfaces, represented as collections of NURBS patches, also used the distance to the model boundary [32]. The main research issues for haptic rendering of sculptured surfaces are initializing to the closest point on the model and stably updating the local closest point. A two-phase approach, a global anticipation phase and local update phase, is a useful model for maintaining high haptic rates [13].

2.1.2 Model-Model Haptic Rendering

While many forms of haptic rendering between user-controlled points and 3D models have been around for the last decade, only recently have reasonable haptic methods for computing model-model interactions become available. In this case, one model is the virtual environment model while the other is the virtual finger, which is represented as a volumetric shape rather than just a point. The primary concern in model-model interactions is computing penetration depth. While minimum distance between the point and model suffices for point-model rendering, a new measure for penetration depth is needed in the general model-model case, as the minimum distance goes to zero when the two models are in collision.

Penetration depth is defined in terms of the vector indicating the minimum distance between two parallel surface tangent planes. These planes are located on each object respectively and are oriented with respect to the local geometry of the interaction region. Figure 2.1 depicts penetration depth for a point-model interaction (left) and a model-model interaction (right).

Haptic rendering algorithms for general polygonal models have approximated the penetration depth by treating portions of the surface as convex patches. The minimum separation distance between patches is then computed and merged into a penetration depth estimate [16]. Johnson and Willemsen avoided this issue by computing all local minima between polygonal models and using these results as forces to push the models away from contact [14]. Sculptured models can use the penetration depth equations

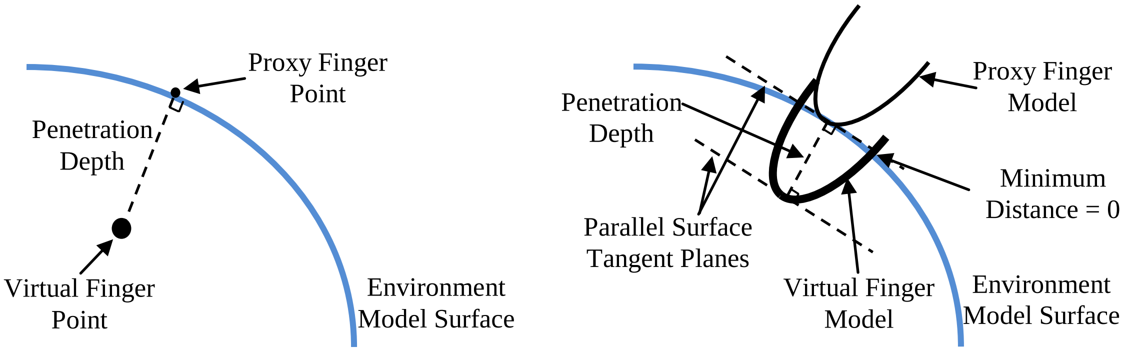


Figure 2.1 Penetration depth shown for a point-model interaction (left) and a model-model interaction (right).

directly. Nelson et al. used an integrable rolling contact formulation for stable updates [24].

Another approach for generalized haptic rendering, which has proven quite successful, precomputes a hierarchical voxel model for the environment and a point sampling of a moving model [21]. Each voxel represents a small cube shaped region of space in the environment. It contains information about its contained space such as penetration depth and force direction. The collection of forces generated between voxels and obstacles defines the force and torque rendered by the haptic interface.

More recently the proxy approach has been generalized to interactions between two polygonal models [22]. A proxy of the virtual finger model is constrained to the surface of the environment model and forces are generated pushing the user-controlled model back to its proxy. This has yielded high-precision rendering on haptic models, even allowing the detection of facets. The algorithms developed in this thesis are designed to run in this generalized proxy method paradigm.

2.2 Combined Tactile and Kinesthetic Feedback

A number of studies have been conducted with combined tactile and kinesthetic feedback. Salada et al. conducted several studies that investigated the use of slip or sliding feedback in combination with kinesthetic motions [29]. Salada was able to show that the addition of sliding feedback allowed users to track small moving features better. The saliency of friction is also increased with slip feedback. Since then, others have also developed slip displays and integrated this with kinesthetic force feedback devices [9, 34]. These devices tend to be large and cumbersome since smaller contact area on the finger relates to weaker sliding cues. Fritschi found that users perceived interactions with slip feedback as more "real". Additionally, Fritschi et al. also investigated providing tactile slip feedback from a tactile pin array in combination with kinesthetic feedback [9]. Again, Fritschi found that providing slip feedback from a pin array increased "realness" 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 [5, 8]. 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 motions to display curved objects [5]. Dostmohamed was able to demonstrate that by providing only normal direction through a gimbaled plate participants were capable of curvature discrimination on par with real life exploration. As a relatively sophisticated adaptation of the this work, Frisoli et al. present a miniaturized finger-based tilting plate tactile display that can be attached to a kinesthetic display [8]. His 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 [18, 26]. The contact location display has been shown to increase awareness of curvature change and edges which enables better contour following. These studies form a precursor to this thesis.

2.3 Haptic Shading Algorithms

Haptic shading algorithms are developed to make polygonal representations of smooth objects feel smooth. Without haptic shading algorithms polygonal models feel rough and textured which detracts from the desired haptic experience.

Polygonal models with haptic shading provide several distinct advantages over using smooth sculpted models defined by explicit surface equations. Because polygonal models consist of discrete points, it is easier to represent arbitrary shapes with triangular polygons than with the explicit equations needed by sculpted surfaces. Additionally, determining interactions between planar objects is far simpler than determining interactions between curved surfaces, which require robust numerical methods that may run too slowly for haptic applications [15]. This is especially important when computing the interactions between the finger model and haptic environment. Lastly, the majority of hardware directly supports the use of polygonal models, facilitating faster interaction times and haptic loops, while sculpted models are not directly supported and are often converted to polygons before graphical rendering.

The most widely used haptic shading algorithm was developed by Morganbesser and Srinivasan [23]. 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, shown in Figure 2.2, to create smooth shadows across polygonal surfaces [25]. Morganbesser and Srinivasan's algorithm was designed to eliminate the “popping” effect felt in rendered normal forces when passing over a vertex or edge of a polygonal object. As with Phong shading, Morganbesser and Srinivasan found their force shading algorithm helped give the sensation of a smoother object.

Other efforts have been directed toward creating surface textures on polygonal objects. Proper texture maps can be used to create smoothed objects or textured contours around objects eliminating unwanted facets. Texture maps are applied by altering the basic force computations as indicated within the texture map. Theoktisto et al. presented a method utilizing height field maps to determine resulting force alterations rather than utilizing traditional vector fields [31]. This method is shown to provide significant sensitivity to the up-down motion when traveling over bumps. However, to utilize this method, proper height maps must be created for each polygon in the model.

Ruspini et al. also incorporated a force shading model which interpolates the normals of the surface [28]. 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 stability issues generated by using the original Morganbesser algorithm when in contact with multiple intersecting shaded surfaces.

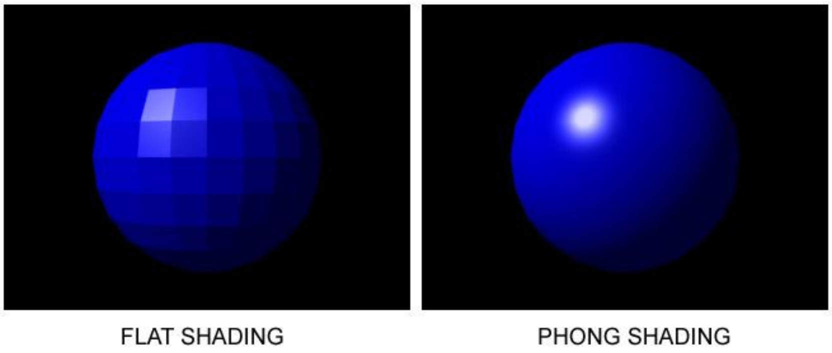


Figure 2.2 Standard polygonal object rendered with both Phong shading (right) and standard flat shading (left). Image courtesy of Wikimedia Commons and is of the Public Domain.

Alternative model representations, like the voxel approach presented by McNeely et al. [21], may be constructed such that haptic shading is already incorporated into the model through the summation of each voxel. In this way small motions make small changes to the active voxels thus creating the effect of a smooth interaction. Unfortunately, approaches like these consume significant computational resources. The use of simple haptic shading algorithms allow a decrease in model resolution thus freeing valuable computation cycles for other tasks.

2.4 Bézier Curves and Surfaces

Both haptic shading algorithms developed as part of this thesis utilize Bézier curves and surfaces to generate smooth interactions. This section provides a brief overview of Bézier curves and surfaces. A more detailed explanation of the Bézier equations necessary for each algorithm is included within their respective chapters.

Bézier curves and surfaces are frequently used in computer graphics as parametric representations of smooth curves and surfaces. These curves and surfaces each are constructed from a control polygon/mesh. Each point in this control polygon/mesh is interpolated with the others to form the resulting curve or surface. For a given number of points, N, the corresponding Bézier curve will be degree (N-1). Thus a control polygon for a curve consisting of three points would result in a quadratic (degree 2) curve. The order of square surfaces is determined independently in each direction along the edge, so a surface of degree 2 in both directions would require a control mesh of nine points, three along each edge. Bézier triangles are an exception to this rule due to their triangular shape. Effectively they wrap the square surface into a triangular form and thus only require six control points to achieve a second order surface but are more difficult to manipulate [6].

2.5 CLD Device Description

The concept for contact location feedback is presented in Figure 1.1, where rather than providing all possible tactile information to a user, only the center of contact is rendered. The hardware utilized in the following experiments consists of a SensAble Phantom Premium 1.5 and a contact location display (CLD) device. The Phantom is used to render environmental 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. This ensures that only the contact position is provided to the user and no skin stretch is experienced. 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 attached to the Phantom via a one-dimensional gimble with sensed tilt angle. The roller is suspended beneath the finger-pad 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 with the user's finger pad.

An open-bottom thimble is used to attach the device securely to users' 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 motor and a 3.175 mm pitch leadscrew 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 Figures 2.3 and 2.4. A close-up view of the fingertip portion of the device is shown in Figure 2.4.

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 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 [26].

2.6 CLD Motion Across Smooth and Faceted Surfaces

The tactile motion of the CLD device depends on the shape of model used. The tactile motion of the CLD device traveling over faceted surfaces in comparison to a smooth curve is demonstrated in Figure 2.5. Note that the contact location remains stationary while traveling across a flat facet, then moves rapidly along the finger while crossing a vertex, whereas the contact location smoothly changes while moving along a curved surface.

This is further emphasized by Figure 2.6 which shows the contact position rendered as a function of motion across a smooth and faceted cylinder. The contact position across the smooth object, shown in green, is nearly a straight line. The small variations in position are due to small angle changes of the user's finger during surface exploration. The collected data using the faceted model show the contact location display moving in bursts rather than the desired smooth line. Again, the data contain a small amount of noise from the exploration. The short horizontal flat sections represent the user's motion over a facet. These are followed by a quick burst of motion as the virtual finger begins to cross an edge. The curved area that follows is a product of the user popping slightly off the surface after passing over the edge.

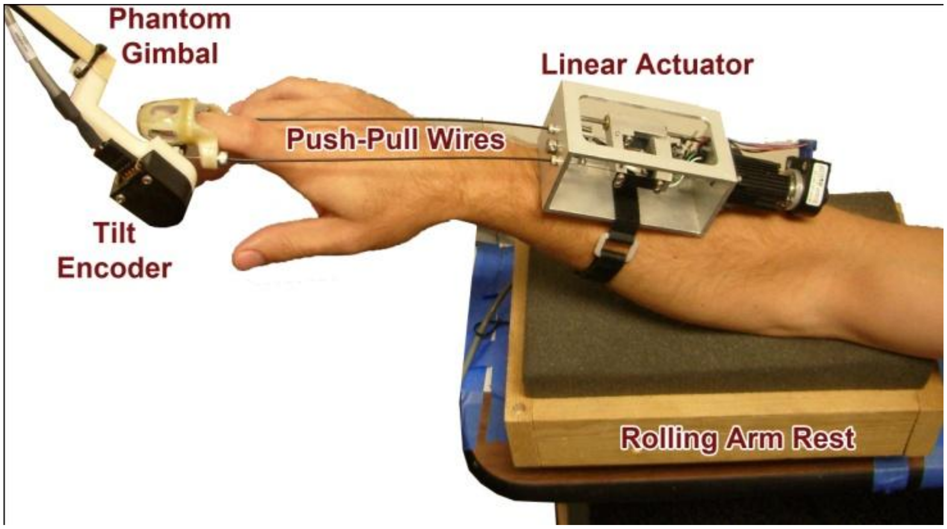


Figure 2.3 Contact location display prototype attached to a Phantom robot. The user's elbow is supported by a rolling armrest.

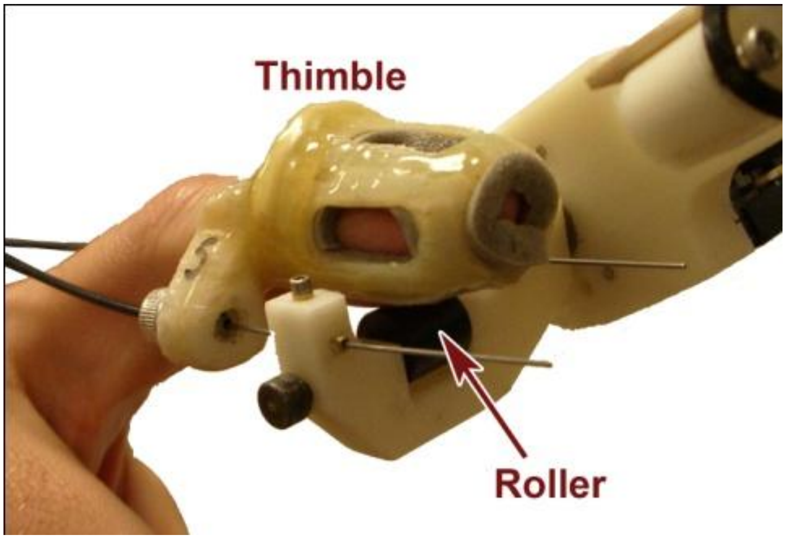


Figure 2.4 The user's finger is secured to the contact location display via an open-bottom thimble.

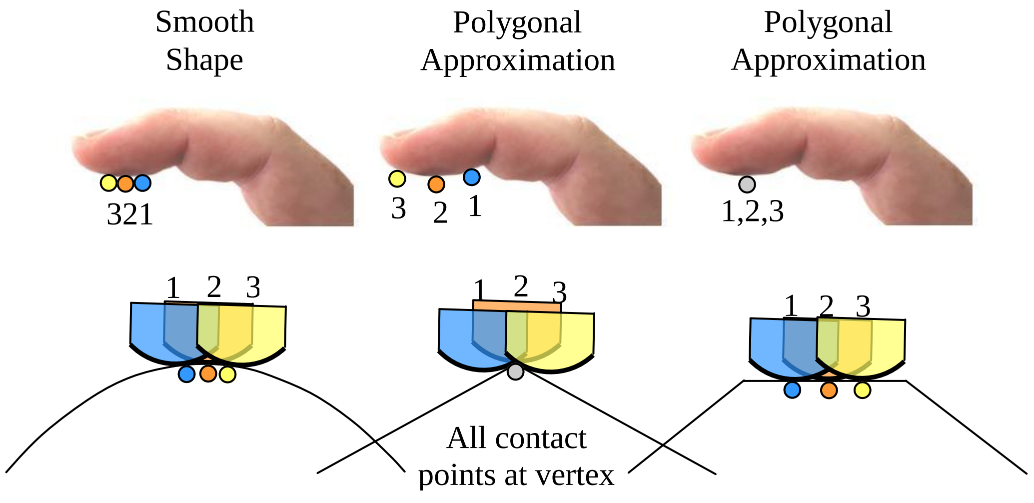
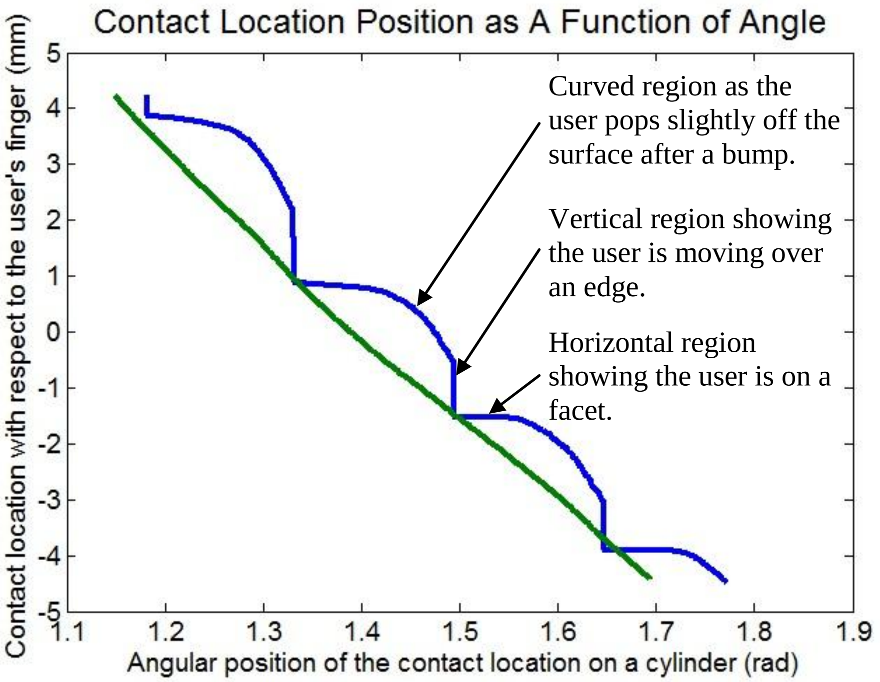


Figure 2.5 Contact location movement over a smooth round surface represented (left) with a curved surface model, (middle) with two facets, and (right) with three facets. The top shows a view of the finger-pad with a series of displayed contact locations, corresponding by color and number to the virtual finger positions below.


Contact Location Position as A Function of Angle

Curved region as the user pops slightly off the surface after a bump.

Vertical region showing the user is moving over an edge.

Horizontal region showing the user is on a facet.

Figure 2.6 Contact location position as the user moves their finger across a smooth surface, shown in green, and a faceted surface, shown in blue. The edges of the faceted model are easily perceived by the user and differ from the ideal smooth model interactions. These data were collected experimentally and contain some noise due to the user's motions.

CHAPTER 3

2D HAPTIC SHADING ALGORITHM

3.1 Overview of the 2D Haptic Shading Algorithm

The shading algorithm developed for 2D polygonal environments utilizes a series of quadratic Bézier curves (refer to Sections 2.4 and 3.2) to create a new smooth curve. This new smooth curve, which makes the underlying facets of the model imperceptible, is used to render (compute and display) the tactile and kinesthetic feedback. Because this shading algorithm is capable of making the facets imperceptible, the number of polygons used can be reduced by a substantial amount while still retaining proper contours. Computing the contact location between two curves requires robust numerical methods that may run too slowly for haptic applications [15]. Instead, the approach computes a dynamically updated tangent line at the point of contact. This tangent is computed directly from the underlying smooth Bézier curve which is never directly computed or interacted with. This allows the continued use of the proxy based methods presented in Section 2.1.2.

Bézier curves have two valuable properties that help define the generated control polygon. These properties are used to guarantee that the resulting surface is smooth and contiguous. First, the end points of the resulting Bézier curve must lie on the control polygon [3]. Second, the tangents of the curve at the end points are equal to the adjacent section of the control polygon [3]. These two properties can be observed in the quadratic Bézier curve, defined by three control points, shown in Figure 3.1.

The shading algorithm first computes the current contact position of the proxy model on the previous iteration's tangent plane. It then uses this contact position to compute a parameter value which is then used to compute this iteration's tangent plane. The cycle then begins again after a short period of time in which the user may have moved their finger. 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 [23] for force shading.

This algorithm was designed to work with two-dimensional polyline objects. All defined vertices must be connected in the polyline which will guarantee the resulting smooth curved surface is also continuous. Multiplicity, or multiple points defined at the same coordinates, can be used to generate sharp corners on the rendered smooth surface.

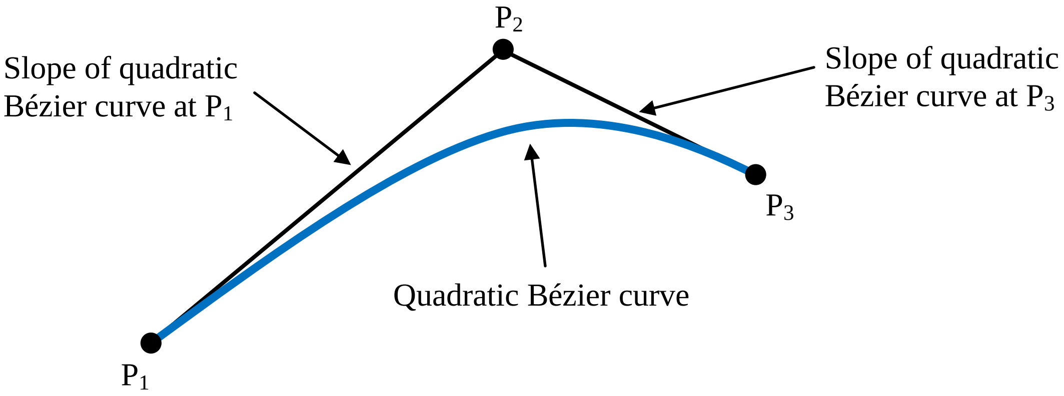


Figure 3.1 A quadratic Bézier curve shown in blue and its control polygon shown in black. The curve's end points are coincident with the control polygon. The tangent of the curve at the end points is equal to the adjacent control polygon points.

Because the algorithm approximates the parameterization from the position of the finger proxy it does not consist of ideal quadratic patches. Instead, the surface is slightly altered but retains both key properties that allow it to join smoothly to neighboring patches. The resulting surface was evaluated for an arbitrary polygonal model shown in Figure 3.2. The black lines represent the original polygonal model and the red, thicker, curved lines represent the shape of the resulting interaction model. The local centers, and their corresponding grey regions, denote separate applications of the algorithm. The resulting curve is built from these patches.

3.2 Bezier Curves

The basis of the 2D haptic shading algorithm is a quadratic Bézier curve. Quadratic Bézier curves are defined by a control polygon containing three ordered points. 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* [3]. Varying the parameter value from zero to one traces out the Bézier curve. In the quadratic case the de Casteljau algorithm is applied twice. Each repetition of the de Casteljau algorithm defines a new set of points containing one fewer point than the previous set. The process is complete when the de Casteljau algorithm returns only one point. In this case the first iteration defines a single line segment. This line segment is tangent to the curve at the point returned in the next iteration. The second iteration defines a point, on that line segment, that lies on the curve. Equations 3.1 are the first iteration of the de Casteljau algorithm. It takes in the three ordered control polygon points (P_1, P_2, P_3) and returns only two points (P_{12}, P_{23}). These points define the line segment tangent to the curve (see Figure 3.3). Equation 3.2 takes the points (P_{12}, P_{23}) from the first iteration and repeats the

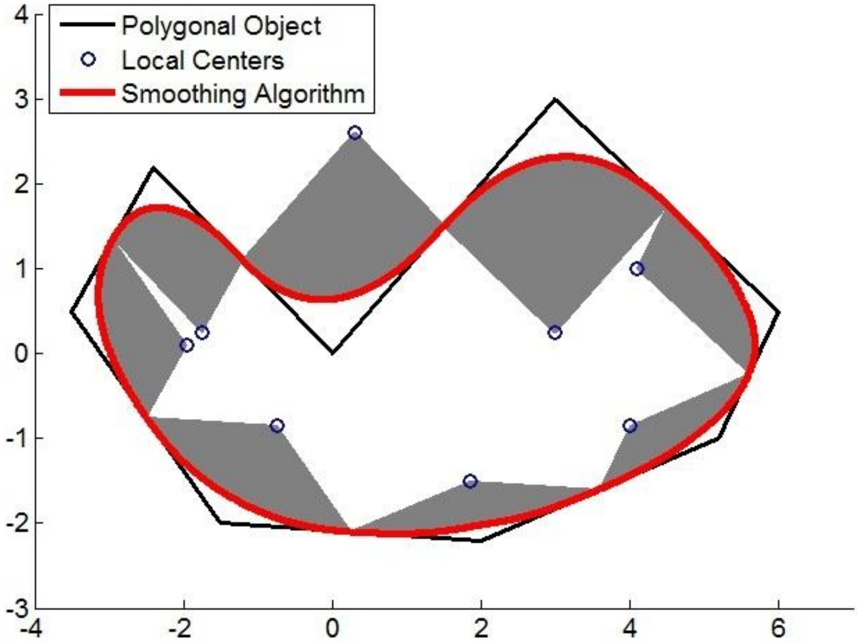


Figure 3.2 The original polygonal model (black) and the smooth interaction model (red). Separate Bézier patches are defined across each region denoted by the grey regions.

de Casteljau algorithm. This returns a single point (P_{1223}) that lies on the curve. The labels used in these equations correlate to those shown in Figure 3.4. The subscripts are used to denote the location of the point.

$$\begin{aligned} P_{12} &= P_1(1-t) + P_2 t \\ P_{23} &= P_2(1-t) + P_3 t \end{aligned} \tag{3.1}$$

$$P_{1223} = P_{12}(1-t) + P_{23} t \tag{3.2}$$

3.3 Defining the Control Polygon

As described in Section 3.1 the 2D haptic shading algorithm utilizes only the tangent plane at the point of contact. Thus only Equations 3.1 are used, and computing the new tangent from a given parameter value is a single step. In order to retain tangent continuity

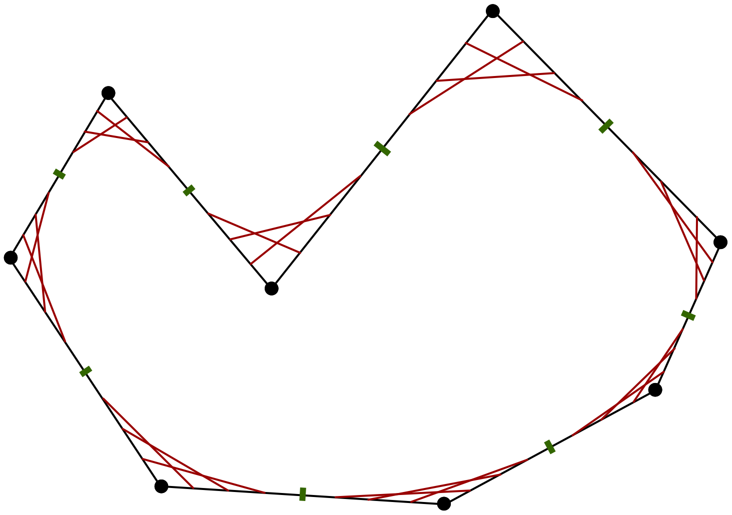


Figure 3.3 An arbitrary polygonal shape. Three tangent line segments are shown for each vertex at *t* = 0.25, 0.50, and 0.75. Only one tangent will be in existence at a single instant in time.

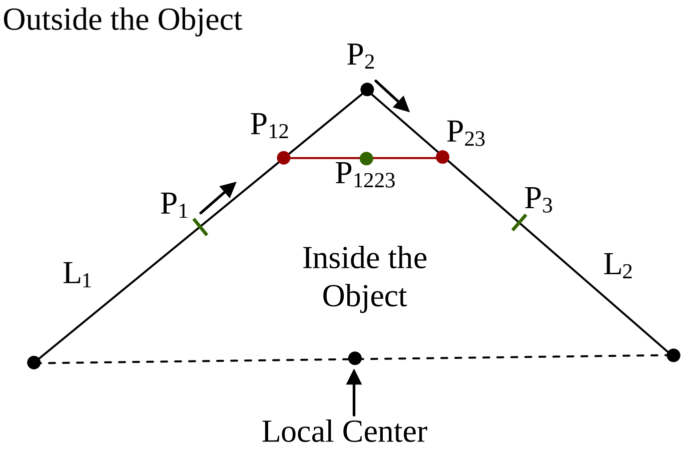


Figure 3.4 Basic labeling scheme used in our shading algorithm.

over patch boundaries, our algorithm forms the Bézier patch from a vertex and the midpoints of each line segment connected to it. Defining the control polygon this way ensures that the resulting surface lies in the convex hull of the original polygonal model and that the surface also contacts the polygonal model at the midpoint of each line segment. Figure 3.3 shows three tangent line segments at *t* = 0.25, 0.50, and 0.75 at each vertex. The adjacent midpoints used are shown as green ticks.

3.4 Implementing the 2D Haptic Shading Algorithm

Before the implementation of the algorithm is described in detail, a few geometric labels must be defined (see Figure 3.4). The two line segments that are adjacent to the vertex of interest are labeled L_1 and L_2. The three control points (P_1, P_2, and P_3) are the vertex and the midpoints of 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. As such 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 normal to L_1 and L_2 placed at their respective midpoints. Placing the local center at the center of curvature 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 3.4. There is only one local center for each vertex.

The next few sections cover each step of the 2D haptic shading algorithm in detail. Pseudo code implementing the algorithm is provided in Appendix A. A brief description of the algorithm is provided here again. The shading algorithm first computes the current contact position of the proxy model on the previous iteration's tangent plane. It then uses this contact position to compute a parameter value which is then used to compute this iteration's tangent plane. This cycle is repeated at haptic rates to give the illusion of a smoothly represented surface.

3.4.1 Computing the Current Proxy Contact Location

To begin each iteration, the proxy contact position is computed using the previous iteration's tangent line as the surface. When moving, this position represents a small differential distance along the tangent line and thus is a reasonable first approximation for determining the user's current contact position on the surface. No forces need to be computed or applied during this step. If the user's finger is not in contact with the surface, the finger is projected onto the surface as normal. At sufficient distance away from the surface the shading algorithm can be entirely skipped and the projection can be based on the original polygonal model.

3.4.2 Computing the New Parameter Value t

To proceed, a parameter value needs to be computed. From this value the de Casteljau algorithm can return the next surface tangent. Finding the parameter value of the Bézier curve that corresponds to the correct contact point on the quadratic curve is difficult and slow. Instead, the parameter value is approximated which slightly alters the resulting surface. One method for finding this parameter value without an explicitly defined curve is nodal mapping [12]. This method relates positions on the defining polygonal model to curve parameter values. However, nodal mapping creates an uneven parameterization that leads to parametric discontinuities while inside of the object. A radial parameterization was instead developed that smoothly and uniformly changes with position and contact location along the curve.

The first step to finding the new parameter value *t* is to determine which vertex 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 the next 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 vertex is used. If constructed carefully, L_1 and L_2 will change at most once in a given iteration.

To better demonstrate this condition check, the current contact point must lie within the shaded region shown in Figure 3.5. If the current proxy contact point is outside of this region, the second condition has been met and needs to be resolved before continuing.

Once L_1 and L_2 have been verified, all that is left is to compute the corresponding parameter value. This is done by computing the angular fraction ($\lambda = \alpha/\beta$) between the current contact point and the closer line segment with respect to the local center. In Figure 3.6 the closer line segment is L_2 and the angular fraction is approximately 0.2. Equations 3.3 and 3.4 show how to calculate the angular fraction for L_1 and L_2 respectively. The points P_0 and P_4, as shown in Figure 3.6, have been added for reference

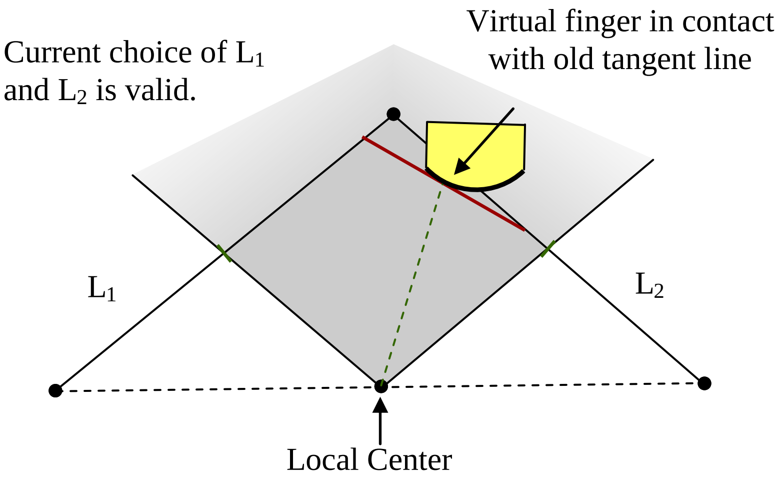


Figure 3.5 Shaded region bounded by the local center and the midpoints of L_1 and L_2 that must contain the computed contact point for L_1 and L_2 to be valid choices.

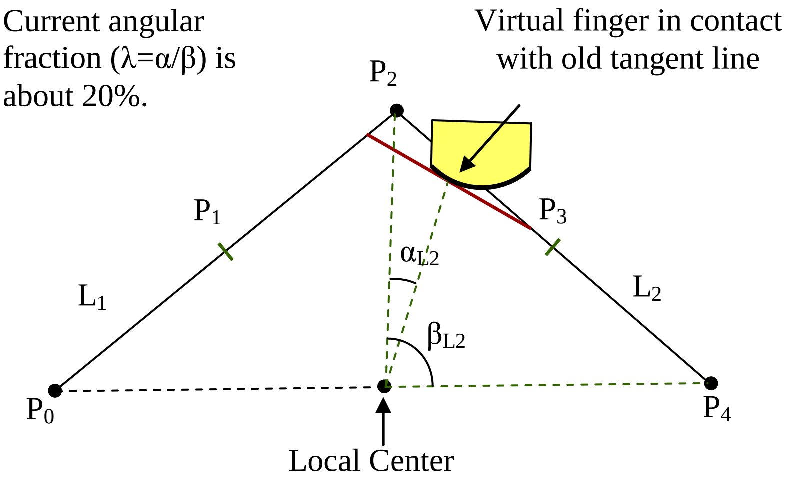


Figure 3.6 Computing the angular fraction based on the active line segment L_2.

reasons and are only used within this step. Note that the angular fraction found when the proxy contact point lies directly between the local center and P_2 will always be either 0 or 1. The angular fractions defined at P_1 and P_3 will be dependent on the chosen position of the local center.

$$\lambda_{L1} = \frac{\alpha_{L1}}{\beta_{L1}} = \frac{\theta_{contact} - \theta_{P0}}{\theta_{P2} - \theta_{P0}} \tag{3.3}$$

$$\lambda_{L2} = \frac{\alpha_{L2}}{\beta_{L2}} = \frac{\theta_{contact} - \theta_{P2}}{\theta_{P4} - \theta_{P2}} \tag{3.4}$$

The angular fraction is then placed into a linear equation to compute the parameter value *t* for the Bézier curve. This equation is linear to guarantee that the reparameterization retains all properties of the original Bézier curve. Because the angular fraction differs when the current contact point projects to L_1 vs. L_2, one of two separate equations is used depending on the closer line segment. These equations must meet three specific criteria for the system to be continuously smooth, or G^1 continuous.

1. t from both equations must be the same for $\lambda_{L1} = 1$ and $\lambda_{L2} = 0$
2. $t = 0$ when the current contact point is at P_1
3. $t = 1$ when the current contact point is at P_3

The first criterion guarantees continuity across the vertex. The second and third criterion forces the resulting curve to 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. These linear equations are more explicitly defined below in Equations 3.5 and 3.6. The t value chosen to be the same for $\lambda_{L1} = 1$ and $\lambda_{L2} = 0$ was 0.5. λ_{P1} and λ_{P3} represent the constant value of λ found at points P_1 and P_3.

$$t = \frac{0.5}{1-\lambda_{P1}}(\lambda_{L1} - \lambda_{P1}) \tag{3.3}$$

$$t = \frac{0.5}{\lambda_{P3}}(\lambda_{L2} - \lambda_{P3}) + 1 \tag{3.4}$$

3.4.3 Computing the New Tangent Line

The last step of this iteration is to update the tangent line segment. This is done simply by plugging the newly computed parametric value into Equations 3.1, shown again below. This returns a line segment tangent to the curve and the next iteration of the shading algorithm begins. However, as the user travels over this line segment they reach its end point (P_1 or P_3) at the midpoint of L_1 or L_2. The user may not be able to pass this point or may fall off the line segment depending on the proxy based haptic rendering method used. To bypass this problem, the line segment can simply be expanded on both ends. This allows smooth switching between line segments over this boundary for all proxy based haptic rendering methods.

$$\begin{aligned} P_{12} &= P_1(1-t) + P_2 t \\ P_{23} &= P_2(1-t) + P_3 t \end{aligned} \tag{3.1}$$

3.5 Evaluation of the 2D Haptic Shading Algorithm

To evaluate the smoothing effects of the 2D haptic shading algorithm, the radius and normals for a series of shaded polygonal approximations to a circle were measured. Since Bézier curves scale along with their control polygon the radius of this circle can be arbitrary, a 100 mm radius was chosen. To evaluate these data the mean and standard deviation of the radius and the angle difference in degrees between the measured normal and its ideal counterpart was computed. The 2D haptic shading algorithm needs to perform well in both the contact position and normal vector categories to create proper sensations. This analysis is presented for each polygonal model tested in Table 3.1. Several things are immediately apparent. As the number of polygons increases, decreasing the angle difference between adjacent polygons, the radius more closely matches the ideal. However, the standard deviations of radius are all very small which implies only a small modification to the model is required to attain the desired radius. The model is nearly indistinguishable from the ideal model when there is around 10° between adjacent facets (as shown in the shaded gray row in Table 3.1). If the model is modified to account for the radius, the proprioceptive cue is removed and much larger angle differences can be used. As expected, the error in angle between the measured and ideal normals is close to zero degrees and decreases as the models more closely match. In all measured models, except the first one, the mean and standard deviation of these normal vector angle errors are less than one tenth of a degree. This is too small to haptically detect. Overall, these data show that even relatively large angle differences between adjacent polygons result in smooth curved surfaces that approximate a circle well.

Table 3.1 Mean and standard deviation of radius and angle error of normal vectors for a variety of angle difference between adjacent polygons evaluated using the 2D haptic shading algorithm.

Angle between adjacent line segments	Mean rendered radius in mm (ideal = 100)	Standard deviation of rendered radius (x10^{-4} mm)	Mean angle error of surface normal in degrees (ideal = 0)	Standard deviation of angle error of surface normal
25.8°	97.5 mm	86.2	0.165	0.681
17.2°	98.9 mm	19.6	0.034	0.023
12.9°	99.4 mm	7.9	0.031	0.021
10.3°	99.6 mm	4.1	0.031	0.020
8.6°	99.7 mm	3.3	0.029	0.018
7.4°	99.8 mm	3.0	0.030	0.020
6.5°	99.8 mm	2.9	0.028	0.018
5.7°	99.9 mm	2.9	0.026	0.018
5.2°	99.9 mm	2.9	0.026	0.017
4.7°	99.9 mm	2.9	0.025	0.016

CHAPTER 4

FIRST EXPERIMENT

4.1 Discrimination Thresholds for Polygonal Smooth Surfaces

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 parametrically 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 [23] 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 [23]. One slight difference from [23] is 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 [23] to a more complete state that can more readily be 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.

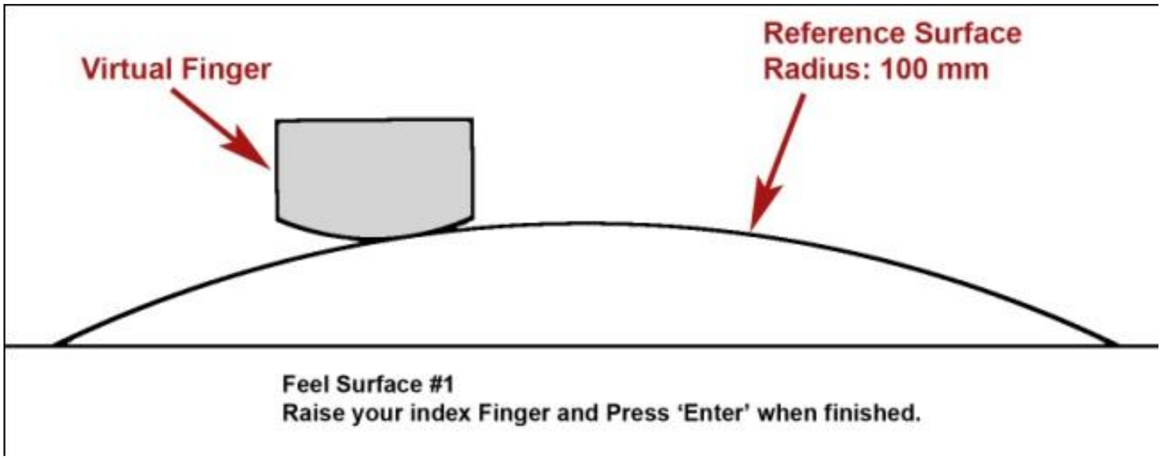
Condition 4 (C4) has participants utilize tactile and kinesthetic feedback to evaluate polygonal models with tactile shading, but without force shading. This condition was designed to evaluate the influence of tactile feedback and can be compared to all three other conditions. The reason that we did 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. That is, subjects had difficulty distinguishing the shaded polygonal and perfectly smooth surfaces when even very few polygons were used, and our adaptive procedure would not be appropriate for evaluating this threshold condition P1 in section 4.3.

4.2 Experimental Methods and Procedures

The experiment utilized a paired-comparison (two interval), forced-choice paradigm, with a 1-up, 2-down adaptive procedure [20]. 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 (making the task more difficult). The threshold obtained corresponds to the 70.7% confidence interval on the psychometric function [20]. The reference stimulus was a mathematically correct arc segment of a circle (see Figure 4.1), 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 section 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.

Each condition was conducted as follows. The participant would first feel stimulus #1. Once they were finished exploring they would then raise their index finger off the surface and press the 'Enter' key to indicate they were ready for stimulus #2. After feeling the second stimulus they would again raise their 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 stimulus presentation was randomized.

The experiment continued until the participant had finished eleven 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. A reduced step size was used for the remaining eight reversals for better accuracy in determining the discrimination threshold. The step sizes chosen for each condition were calibrated during



Feel Surface #1
Raise your index Finger and Press 'Enter' when finished.

Figure 4.1 Screen capture of the smooth reference object used during training that preceded each test condition.

pilot testing and fixed for all participants in the study. The discrimination thresholds were computed from the last six reversals.

A Latin Squares reduction of the system was utilized to reduce the number of permutations for balancing testing order in which participants completed the four experimental conditions. The testing apparatus, as shown in Figure 4.2, 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 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.

Twelve right-handed individuals (three females) between the ages of nineteen and forty-one participated in the experiments. They took an average of about ten minutes and

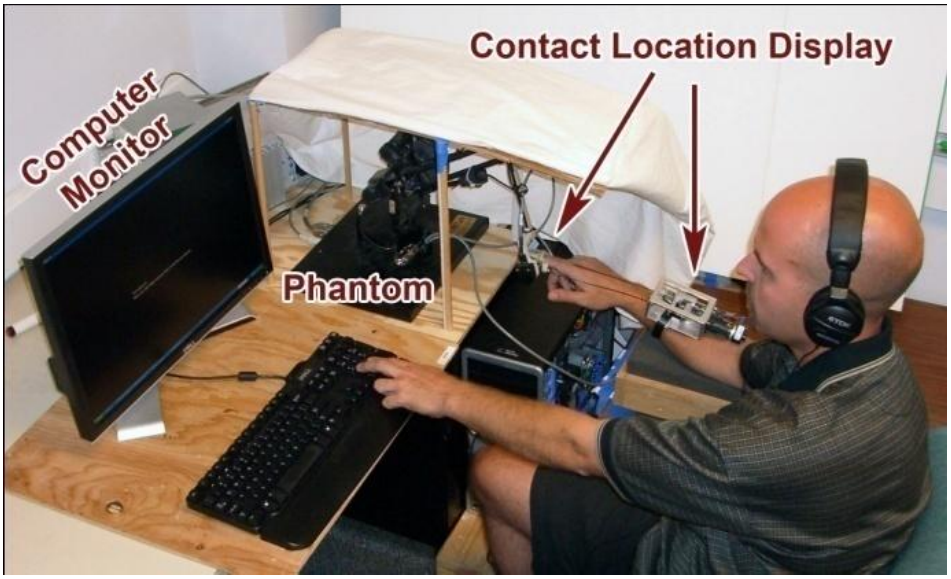


Figure 4.2 Experiment test setup (cover pulled back for clarity)

forty-two trials to complete each condition. Additionally, no learning effects due to the ordering of the four conditions were observed.

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 below in terms of the more general metric of the angle difference between adjacent polygons.

4.3 Experiment Results

Two representative data sets for one participant are shown in Figure 4.3. Note that this participant had some difficulty in Condition 2 (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 their performance before completing the eleven reversals.

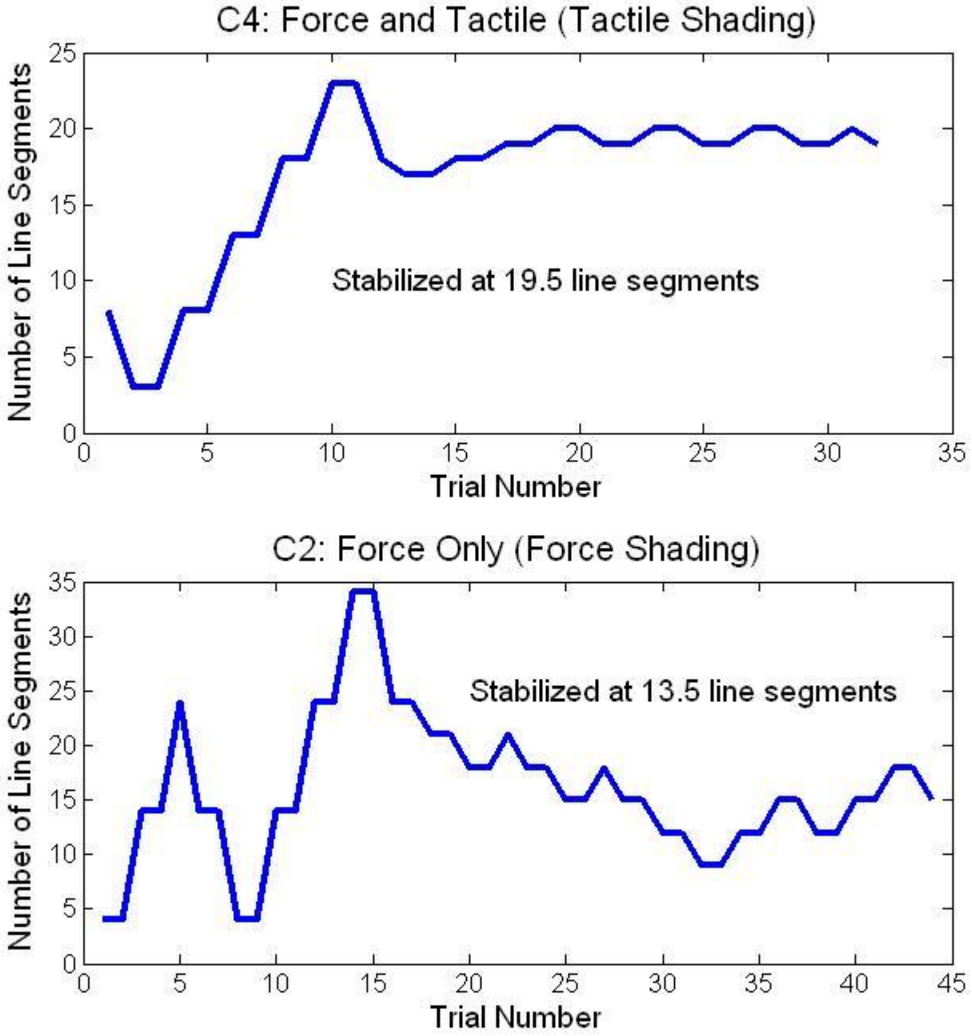


Figure 4.3 Two collected data plots showing (top) nearly ideal data from one participant and (bottom) less ideal data from the same participant who had difficulty with C2.

The data collected from the twelve participants passed an omnibus ANOVA test ($F(44,47) = 47.76$, $p < 0.001$). This implies independence between all four conditions and allows the use of Tukey's test to determine if the results are significantly different. The data were subsequently analyzed for statistically significant differences using Tukey's test with $\alpha = 0.05$. 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 than all other conditions. The two shading conditions (C2 and C4) were not significantly different from each other. Table 4.1 shows the mean discrimination thresholds and the corresponding 95% confidence intervals for the collected data. To best understand the practical implications of this data, it is useful to consider this example. If the angle difference between adjacent polygons in a model used is less than the lower end of the 95% confidence interval (for example less than 0.37° for C1) then 97.5% or more of people should sense the model as perfectly smooth. Note that the subjects were concentrating on the smoothness, so for tasks involving distractions, these angle thresholds would increase. Figure 4.4 plots these means and confidence intervals, and helps highlight the significant differences among the four conditions.

Table 4.1 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	C2	C3	C4
	Force Only	Force Only with Force Shading	Force and Tactile	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°)

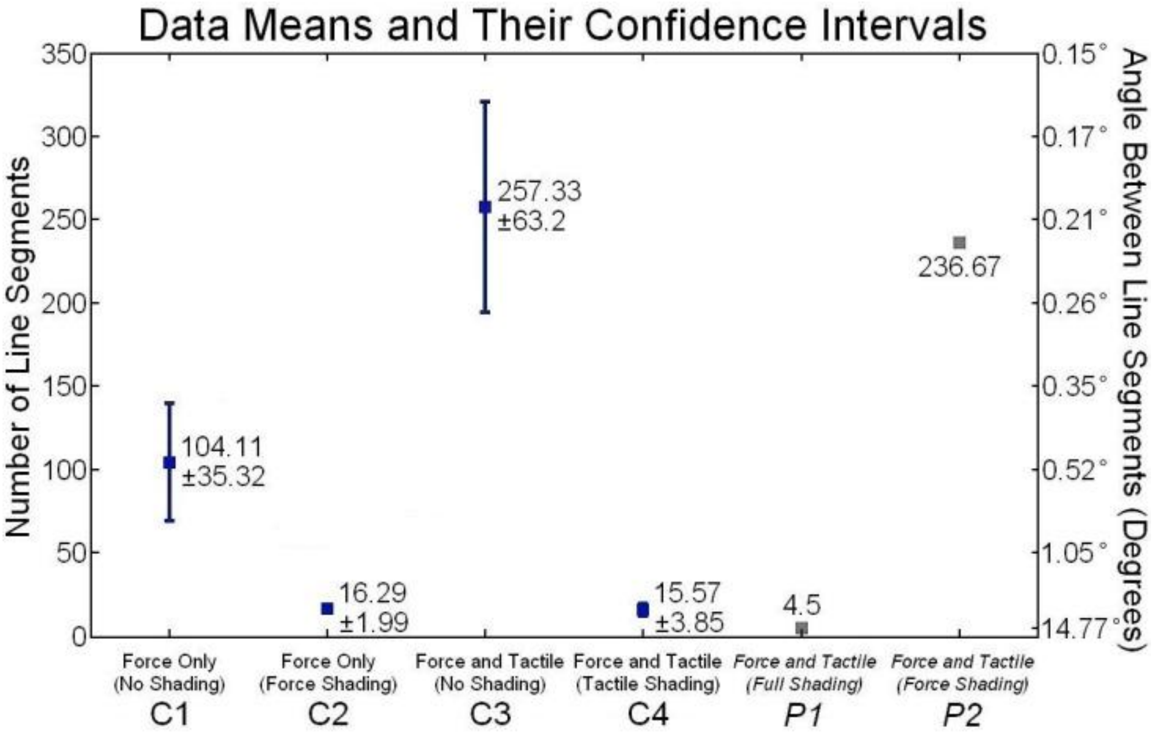


Figure 4.4 Plot of the mean and 95% confidence intervals for each test condition showing the number of line segments at which the polygonal model was indistinguishable from the smooth reference surface. The error bars are not linear when interpreting results based on the angle difference between segments.

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 [23] 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 changes in the physical shape itself, e.g., angle differences between adjacent polygons. Table 4.1 shows the angle difference thresholds corresponding to the line-segment thresholds in parentheses. The same angle differences are shown in Table 4.2 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.

Table 4.2 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)

4.4 Discussion

First, we compare the measured threshold for C1 to prior work. Our results are not directly comparable to that of Morganbesser and Srinivasan, 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 discriminating the angle difference between sequentially applied force vectors. Barbagli et al. report 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°) [2]. This is not surprising though as people have much greater sensitivity to changes presented in rapid succession [10]. Our task also utilized active rather than passive sensing in making perceptual judgments, which is also expected to provide greater perceptual sensitivity [17].

Several trends can be observed from the data presented in Tables 4.1 and 4.2. First of all, the addition of tactile feedback greatly increases one's sensitivity to edges and vertices in the system, as seen by a pair-wise comparison of the thresholds for C1 and C3 and those for C2 and P2 in Table 4.2. This increased sensitivity is undesirable when smooth surfaces are rendered because more line segments are needed in order to render a smooth surface with a polygonal model, 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 shading algorithm, presented in Chapter 3, significantly reduce the needed number of line segments to make a polygonal object feel smooth. Note that the shading algorithm developed in Chapter 3 also is capable of rendering force shading and can therefore reduce the number of line segments further as indicated by the threshold of 14.8° for P1 shown in Table 4.2.

Another interesting observation is that when both force and tactile feedback signals are present, people appear to rely more on tactile than force information to judge the smoothness of a surface. If the participants completely ignored the tactile sensations, then there should be no difference between the thresholds for C1 and C4 in Table 4.2. Instead, the 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 haptic perception than the force irregularities. In fact, in the presence of unshaded tactile information (C3 and P2), there appears to be no significant benefit from applying Morganbesser and Srinivasan's force shading algorithm in P2 (see Table 4.2).

Utilizing both force and tactile shading can significantly reduce the required polygonal model size further, as is shown for P1 in Table 4.2. This condition was not evaluated in our main experiments because there were not enough stimulus levels (in terms of line segments in a polygonal model approximating a smooth surface) to implement the adaptive procedure. While the minimum approximating model size would be two line segments, it took at most five segments before the polygonal model became indistinguishable from the ideal shape. This could be because our shading algorithm also more closely approximates the ideal shape and size of the original smooth object in our test conditions, hence also eliminating proprioceptive cues from being discerned.

To summarize, the use of shading algorithms can lead to a significant reduction in the size of polygonal models approximating smooth object surfaces without introducing noticeable artifacts. Referring to the angle difference thresholds for C1-C4 in Table 4.2, it is shown that the addition of shading allows angle differences between adjacent polygons to reach ~3° before any nonsmoothness is perceived as opposed to only 0.2-0.5° when shading is not used. Furthermore, our pilot tests (P1) indicate that polygon models with as much as 15° between adjacent polygons are perceived to feel smooth while still properly representing object shape and size if our smoothing algorithm was utilized to apply both tactile and force shading. 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, the angle difference thresholds are likely applicable to other types of tactile displays including those that render the tangent planes of a curved surface [5, 7].

4.5 Experiment Conclusions

The addition of tactile feedback has the potential to increase immersion and dexterous capability with haptic systems. Our experiments show that the addition of tactile feedback significantly increases one's sensitivity to discontinuities in the environment and provides a new mode to gain haptic information. Because of this increased sensitivity, it may be necessary to smooth out unintended modeling artifacts that will be present when interacting with polygonal models – especially when large numbers of polygons are utilized in order to keep model size tractable. To eliminate these tactile artifacts and improve immersion, a tactile shading algorithm was developed and presented. The effect of the 2D haptic shading algorithm on the way polygonal models are perceived was evaluated alongside more traditional force shading. This algorithm was determined to reduce the required haptic model size significantly while still allowing greater immersion.

We have also investigated the perceptibility of surface discontinuities by comparing the use of force feedback alone to combined force and tactile feedback with and without the respective shading algorithms. Very small angle differences between adjacent polygons (0.2-0.5°) were required 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 4.4 and Table 4.2. Either form of force or tactile shading allowed a relatively large angle difference between polygons (~3°, a factor of 6), while greater angle differences between polygons (~15°, a factor of 30) were possible if both force and tactile shading was simultaneously applied, thereby requiring a significantly smaller number of polygons to represent a given haptic model.

CHAPTER 5

3D HAPTIC SHADING ALGORITHM

5.1 Overview of the 3D Haptic Shading Algorithm

A 3D haptic shading algorithm for polygonal models was developed to build upon the previously presented shading algorithm for 2D environments. As with the 2D haptic shading algorithm in Chapter 3, each element of the polygonal model is used to generate a control mesh. This control mesh defines a triangular curved surface via Bézier triangles. However, Bézier triangles alone are incapable of creating a surface with G^1 continuity, yet G^1 continuity is needed to guarantee that the surface feels smooth. G^1 continuity means that the direction of the tangent to the surface is continuous. The tangent vector instantaneously changes direction when crossing an edge. Thus, without G^1 continuity, the surface retains edges which may still be noticed. While it is possible to fit smooth surfaces to polygonal models the process is difficult and time consuming [3, 4]. Instead, this continuity is locally added through the use of quadratically interpolated normals to create a sense of G^1 continuity.

A means of computing the control meshes for the Bézier triangles and quadratically interpolated normals was adapted from the computer graphics literature. This process was originally introduced by [33] as PN (point normal) triangles. PN triangles are a method for producing 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 their local smoothing processes independently of the other triangles in the mesh. This can be taken advantage of to increase the haptic rendering efficiency of the shading algorithm. As with the 2D haptic shading algorithm the 3D haptic shading algorithm renders only the tangent plane at the point of contact. The rendered surface then becomes defined by a composite plane generated by the point computed by the Bézier triangle surface and normal defined by the quadratic interpolation. This shading algorithm is also capable of being used in place of the methods presented by Morganbesser and Srinivasan [23] for force shading.

The process the 3D haptic shading algorithm takes is similar to the 2D haptic shading algorithm presented in Chapter 3. The 3D haptic shading algorithm first computes the current proxy's contact position on the previous iteration's tangent plane. It then uses this point to compute the parametric values for the following iteration. The new tangent plane is then computed from these new parametric values and the process is repeated. Since attempts to find an inverse equation relating position to parametric values proved too difficult, numerical methods were instead used to gain convergence on the ideal contact point. Thus within each haptic rendering cycle (a minimum of 1000 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. Thus the number and speed of these iterations needs to be small enough to be completed within the 1 ms time frame of each haptic rendering cycle.

This algorithm was designed to be utilized on a well formed polygonal mesh, without any floating edges or triangles. By ensuring the model is fully connected we can guarantee the resulting surface from the 3D haptic shading algorithm is continuous and smooth. To facilitate fast rendering times each triangle in the mesh also contains information on the three other triangles that share its edges.

As with the 2D haptic shading algorithm, multiplicity can be used to create corners and edges. In this case multiplicity implies the addition of multiple normals on a single vertex. This can also be done by defining extra triangles with zero width. If the additional normal vectors at the vertices are not perpendicular to the straight edge being defined, the equations used to generate the PN triangle surface will no longer match on either side of the edge. Effectively a hole in the surface will form along this edge resulting in a noticeable discontinuity. It is advised instead to add extra triangles in these regions to more definitively define the feature.

5.2 PN Triangles

This section provides a brief description of the methods developed in the PN triangles paper and provides the equations necessary to compute the curved surface and corresponding normals [33].

5.2.1 Defining the Control Mesh

The control mesh used to compute the surface position of PN triangles is defined by ten points. This creates a 3rd order surface in all three parametric directions (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). 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 5.1 shows a base triangle in blue and its generated control mesh. The three outer most triangles of the control mesh each are defined such that they share a point on the base triangle and the normal at that point. 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.

The mesh points are computed in three stages. The first stage contains three points which are the initial base triangle points. These are indicated in Figure 5.1 as b_{300}, b_{030}, and b_{003}. The second stage consists of the six points around the center point with a 2 in their indices. For each of the three points in the prior stage another point is computed one-third of the way down each of the adjacent base triangle's sides. This point one-third down the side is then projected onto the plane defined by the corresponding point from the base triangle and its normal. The resulting intersection is the control mesh point. Figure 5.2 demonstrates this process for computing b_{210}. The final stage, computing b_{111}, is done by computing the center of each of the previous two stages separately and determining the vector from the center of the base triangle to the center of the six new control points. The point b_{111} is then one and a half times this vector from the center of the base triangle. The control mesh generated with these methods causes the smooth shaded surface to bulge away from the polygonal model. However, the smooth shaded surface is attached to the polygonal model at each of its vertices through b_{300}, b_{030}, and b_{003}. Equations 5.1 show exactly how to compute all ten of the control mesh terms from P_1, P_2, P_3, N_1, N_2, and N_3 [33].

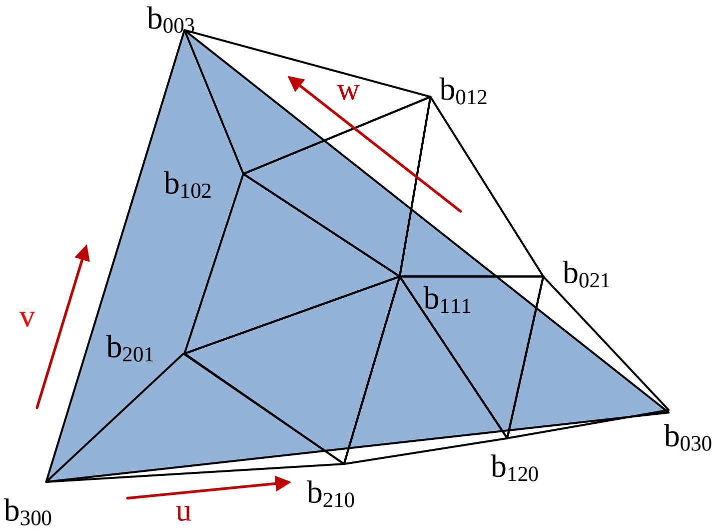


Figure 5.1 A control mesh generated for a particular base polygon. The mesh is defined completely by the three normals defined at each of the three vertices on the base polygon and their relationships. The red vectors represent the directions of the barycentric coordinates u, v, and w used as parametric inputs.

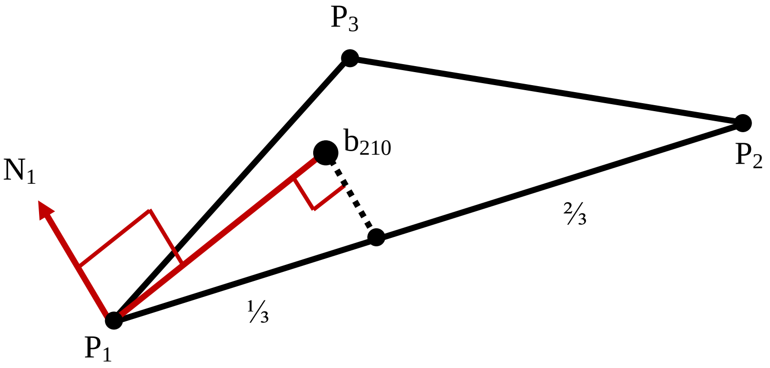


Figure 5.2 Computing the control mesh points. The second stage of control points containing 2 in their indices are created by projecting a point on the edge of the base triangle onto the plane defined by the vertex point and its normal.

$$
\begin{aligned}
b_{300} &= P_1 \\
b_{030} &= P_2 \\
b_{003} &= P_3 \\
b_{210} &= \frac{1}{3}\left(N_1\left(N_1 \cdot \left(P_2 - P_1\right)\right) + 2P_1 + P_2\right) \\
b_{201} &= \frac{1}{3}\left(N_1\left(N_1 \cdot \left(P_3 - P_1\right)\right) + 2P_1 + P_3\right) \\
b_{120} &= \frac{1}{3}\left(N_2\left(N_2 \cdot \left(P_1 - P_2\right)\right) + 2P_2 + P_1\right) \\
b_{021} &= \frac{1}{3}\left(N_2\left(N_2 \cdot \left(P_3 - P_2\right)\right) + 2P_2 + P_3\right) \\
b_{102} &= \frac{1}{3}\left(N_3\left(N_3 \cdot \left(P_1 - P_3\right)\right) + 2P_3 + P_1\right) \\
b_{012} &= \frac{1}{3}\left(N_3\left(N_3 \cdot \left(P_2 - P_3\right)\right) + 2P_3 + P_2\right) \\
E &= (b_{210} + b_{201} + b_{120} + b_{021} + b_{102} + b_{012}) / 6 \\
V &= (b_{300} + b_{030} + b_{003}) / 3 \\
b_{111} &= E + (E - V) / 2
\end{aligned}
\tag{5.1}
$$

The control mesh for the quadratically interpolated normals contains only six points to define a second order system. Again, the first stage of points are from the base polygonal triangle. They are set to the normals associated with the corners (N_1,N_2,N_3). The computations for the first and second stages are provided below as equations 5.2 [33]. The specific format of these equations helps guarantee that there is inflection in the computed normals where there is one on the Bézier triangle surface. Since the control mesh for computing the quadratically interpolated normals is constructed of normal vectors, all its vectors must be normalized to 1 before being used.

$$
\begin{aligned}
n_{200} &= N_1 \\
n_{020} &= N_2 \\
n_{002} &= N_3 \\
n_{110} &= normalized\left[N_2 + N_1 - 2\frac{(P_2 - P_1)\cdot(N_2 + N_1)}{(P_2 - P_1)\cdot(P_2 - P_1)}(P_2 - P_1) \right] \\
n_{101} &= normalized\left[N_1 + N_3 - 2\frac{(P_1 - P_3)\cdot(N_1 + N_3)}{(P_1 - P_3)\cdot(P_1 - P_3)}(P_1 - P_3) \right] \\
n_{011} &= normalized\left[N_3 + N_2 - 2\frac{(P_3 - P_2)\cdot(N_3 + N_2)}{(P_3 - P_2)\cdot(P_3 - P_2)}(P_3 - P_2) \right]
\end{aligned}
\tag{5.2}
$$

5.2.2 Computing the Barycentric Coordinates

PN triangles uses barycentric coordinates, which are commonly used to define positions on triangles in terms of u, v, and w, as its parametric coordinates. They are a system of homogenous coordinates based on the signed areas of the base triangle and the sub triangles formed by the target point. Figure 5.3 (left) shows the position and value of some basic barycentric coordinates. The u, v, and w barycentric coordinates of the point shown in Figure 5.3 (right) can be computed by taking the signed area of the sub-triangles A, B, and C then dividing by the area of the overall triangle.

5.2.3 Computing the PN surface

While the de Casteljau algorithm does exist for Bézier triangles it is faster to compute the single resulting surface point directly in this case. Equation 5.3 uses the control mesh as labeled in Figure 5.1 along with the barycentric coordinates u, v, and w to compute the surface point [33]. Equation 5.4 is used to compute the quadratically interpolated normal vector for the same barycentric coordinates [33].

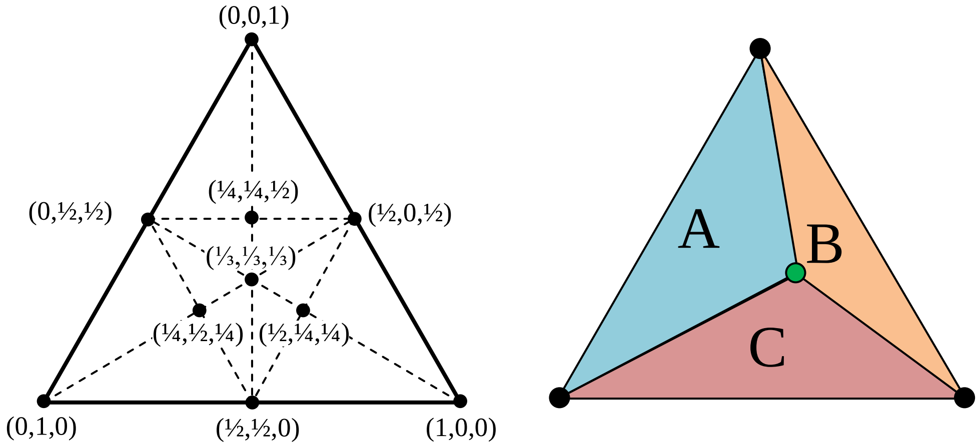


Figure 5.3 Barycentric coordinates computed from a triangle. Left common barycentric coordinates. Right Finding the barycentric coordinates is done by computing the signed areas of A, B and C divided by the area of the overall triangle.

$$\begin{aligned} P &= b_{300}w^3 + b_{030}u^3 + b_{003}v^3 \\ &+ 3(b_{210}w^2u + b_{120}wu^2 + b_{201}w^2v + b_{021}u^2v + b_{102}wv^2 + b_{012}uv^2) \\ &+ 6(b_{111}wuv) \end{aligned} \tag{5.3}$$

$$N = n_{200}w^2 + n_{020}u^2 + n_{002}v^2 + 2(n_{110}wu + n_{011}uv + n_{101}wv) \tag{5.4}$$

5.3 Implementing the 3D Haptic Shading Algorithm

This section provides detailed descriptions of each step taken in the algorithm. Pseudo code implementing the algorithm is provided in Appendix A. A brief description of the algorithm is as follows. The 3D haptic shading algorithm first computes the current proxy's contact position on the previous iteration's tangent plane. It then uses this point to compute the next parametric values for the following iteration. The new tangent plane is then computed from these new parametric values and the process is repeated. This process may be repeated several times in a single haptic rendering cycle until the ideal contact point is reached, that is, until the proxy's contact location is the point generated by the Bézier triangle surface.

5.3.1 Computing the Current Proxy Contact Location

In this step, the current position of the user is orthogonally projected toward the tangent plane from the previous iteration into contact. Since a sphere model for the virtual finger is being used this is a projection that moves the current center of the virtual finger in the direction of the tangent plane's normal such that it is one radius away. The difference between the previous ideal contact point and the current contact position of the proxy defines a direction of travel. The properties of the PN triangle surface guarantees this direction vector applied to the base triangle directly relates to the direction of the barycentric coordinates that will result in a closer point and normal. Depending on the direction of the computed surface normal vectors, this direction may not form a straight path to the ideal contact point.

5.3.2 Computing the New Parameter Value

The direction vector found in the previous step is used to compute a new set of barycentric values. The direction vector is first scaled by a gain that decreases as the local surface curvature and absolute distance from the surface increase. The inclusion of this gain substantially improves the stability and convergence of the system across a variety of object models. Since each of the two terms, surface curvature and absolute distance, ranges from zero to infinity, the terms are increased by 1 before division. This guarantees that the scaled direction vector is always smaller in magnitude than the original direction vector. Equation 5.5 shows the computation of this gain where G_k and G_d are positive factors relating the importance of each term and are set by the user (k is the curvature, and d is the distance from the tangent plane).

$$Gain = \frac{1}{(1+G_k k)(1+G_d |d|)} \tag{5.5}$$

The distance from the surface (d) is defined as the distance from the current position of the user to the proxy model on the surface of 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 smaller parametric steps are taken.

Because the surface is a composite between the Bézier triangle surface and quadratically interpolated normals the curvatures cannot be computed directly via normal methods. Given the previous contact point's normal and a travel direction a plane can be defined that contains these two vectors. The intersection between this plane and the curved surface defines a space curve. The curvature in the travel direction can then be computed from this space curve. Since this space curve is not likely to be arc length parameterized, 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 5.6). 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 (N) and the tangent vector (T) are always orthogonal, the magnitudes of their derivatives are also equal. Because we have separate equations for the position (s) and normal vector (N) in our 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) (see Equation 5.6).

$$k = \frac{\|dT\|}{\|ds\|} = \frac{\|dN\|}{\|ds\|} \tag{5.6}$$

The derivatives that define curvature are relatively simple to compute using the chain rule. First the primary variable u or v is determined using the scaled direction vector. The variable chosen should be the one most parallel to the scaled direction vector. This variable will define the derivatives of u, v, and w. The derivative of the primary variable is always 1. The derivative of the remaining variable, u or v, is set to the change in that variable over the change in the primary variable as defined by the scaled direction vector. The derivative of w is always negative the sum of the derivatives of u and v. Equations 5.7 should be used when v is the primary variable. Equations 5.8 should be used when u is the primary variable.

$$\begin{aligned} \dot{v} &= 1 \\ \dot{u} &= \frac{u_2 - u_1}{v_2 - v_1} \\ \dot{w} &= -\dot{u} - \dot{v} \end{aligned} \tag{5.7}$$

$$\begin{aligned} \dot{u} &= 1 \\ \dot{v} &= \frac{v_2 - v_1}{u_2 - u_1} \\ \dot{w} &= -\dot{u} - \dot{v} \end{aligned} \tag{5.8}$$

Next the partial derivatives of position with respect to u, v, and w are computed (see Equation 5.9). The derivative of position with respect to the primary variable is then computed using chain rule composition (see Equation 5.10).

$$\begin{aligned}\dot{P}_w &= 3(b_{300}w^2+b_{120}u^2+b_{102}v^2)+6(b_{210}wu+b_{201}wv+b_{111}uv)\\ \dot{P}_v &= 3(b_{003}v^2+b_{201}w^2+b_{021}u^2)+6(b_{102}wv+b_{012}uv+b_{111}uw)\\ \dot{P}_u &= 3(b_{030}u^2+b_{210}w^2+b_{012}v^2)+6(b_{120}wu+b_{021}uv+b_{111}vw)\end{aligned} \tag{5.9}$$

$$ds = \frac{d}{d(u\ or\ v)}\dot{P} = \dot{P}_w\dot{w} + \dot{P}_u\dot{u} + \dot{P}_v\dot{v} \tag{5.10}$$

All that remains is to compute the derivative of the normal vector. Since the normal vector is divided by its magnitude the resulting chained form equation is slightly more complicated. Firstly, N and its derivative are computed (see Equations 5.11, 5.12, and 5.13). Then the derivative of the unit normal with respect to the primary variable is computed using equation 5.14. Finally, the curvature can be computed as shown in equation 5.6.

$$\begin{aligned}\dot{P}_w &= 3(b_{300}w^2+b_{120}u^2+b_{102}v^2)+6(b_{210}wu+b_{201}wv+b_{111}uv)\\ \dot{P}_v &= 3(b_{003}v^2+b_{201}w^2+b_{021}u^2)+6(b_{102}wv+b_{012}uv+b_{111}uw)\\ \dot{P}_u &= 3(b_{030}u^2+b_{210}w^2+b_{012}v^2)+6(b_{120}wu+b_{021}uv+b_{111}vw)\end{aligned} \tag{5.11}$$

$$\begin{aligned}\dot{N}_w &= 2(n_{200}w+n_{110}u+n_{101}v)\\ \dot{N}_v &= 2(n_{002}v+n_{101}w+n_{011}u)\\ \dot{N}_u &= 2(n_{020}u+n_{110}w+n_{011}v)\end{aligned} \tag{5.12}$$

$$\dot{N} = \dot{N}_w\dot{w} + \dot{N}_u\dot{u} + \dot{N}_v\dot{v} \tag{5.13}$$

$$dN = \frac{d}{d(u\ or\ v)}\left(\frac{N}{\|N\|}\right) = \frac{\dot{N}\|N\| - N\dfrac{N \cdot \dot{N}}{\|N\|}}{\|N\|^2} \tag{5.14}$$

Once the direction vector is scaled by distance and curvature (see Equation 5.5) 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.

5.3.3 Computing the New Tangent Plane

After the new barycentric coordinates have been computed the error needs to be evaluated. First the new surface point and normal are computed. These will form the basis for the starting point of the next iteration. Ideally, the user's current position projected orthogonally into contact with the tangent plane would be contacting at the new surface point. Because the used normals are a vector field independently defined from the surface points this ideal interaction point may not necessarily be the closest point on the Bézier surface to the user's position. If the closest point on the Bézier surface is instead used it can cause discontinuities at triangle boundaries due to the lack of G^1 continuity of

Bézier triangles. The comparison between the two projection methods is shown below in Figure 5.4. Here the user's finger follows the orange path and is projected to the surface. If projected in the direction of the surface normal a discontinuity forms at the connecting edge of two Bézier triangles. If the computed normal vector is used instead the projections to the surface are continuous and smooth.

If the distance between the proxy contact point and computed Bézier surface point is too large (> 1 µm) the process is repeated again using the newly computed tangent plane. The system takes on average two to three iterations to converge. Figure 5.5 demonstrates this iterative process for a sample two-dimensional cross-section of a shaded surface. The user's finger may be located anywhere inside or outside the object. The computed proxy contact location is denoted by the gray filled circle and the projection direction is shown as an arrow from the finger. The previous iteration is drawn in grey while the current iteration's tangent plane is marked in black. In the final iteration, the computed proxy contact location finally converges close enough to the previous iteration's surface point. The iterations then stop and the previous iteration's tangent plane is used as the rendered surface.

5.4 Evaluation of the 3D Haptic Shading Algorithm

To evaluate the smoothing effects of the 3D haptic shading algorithm the radius of curvature for a series of polygonal approximations to a sphere was measured. As with the 2D haptic shading algorithm Bézier surfaces are proportional to their control mesh. A 100 mm radius was chosen again for easy comparison. Several polygonal models, each with a different angle difference between adjacent polygons, were evaluated with the shading algorithm. The means and standard deviations of the radius and angle difference

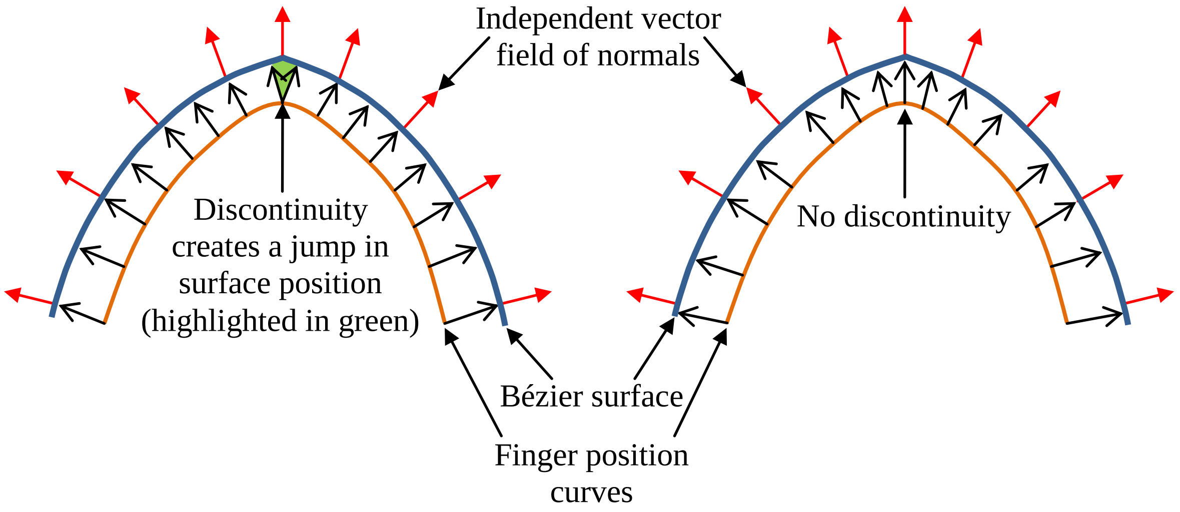


Figure 5.4 Projection of a series of points inside the surface using the surface normal (left) and using the independently computed vector field of normals (right)

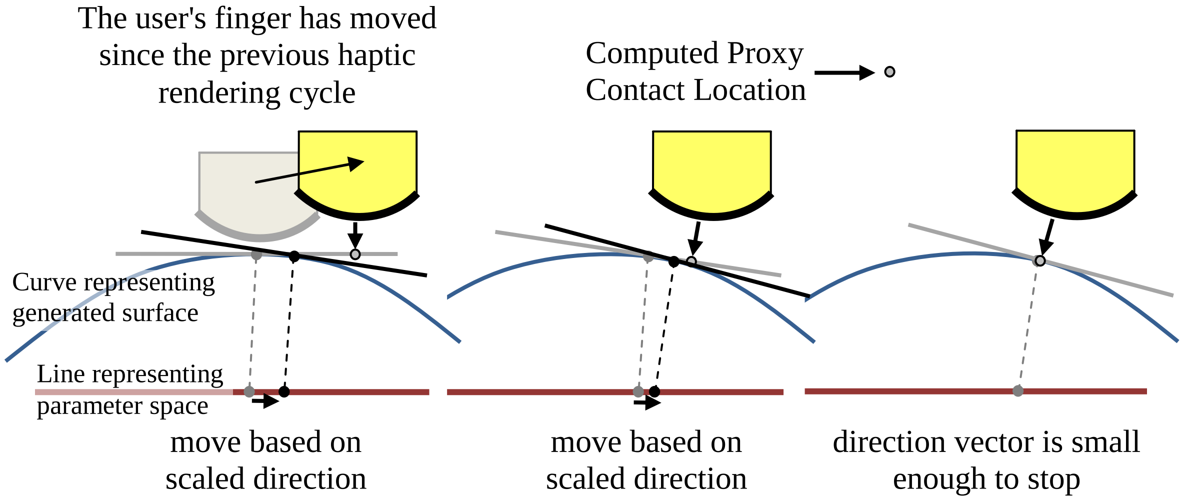


Figure 5.5 The tangent plane marches toward the ideal contact point where the proxy contact point is the rendered surface point. This is drawn chronologically from the left to the right. The system converges quickly. The previous iteration is shown in gray.

of the normal from ideal are reported in Table 5.1. The 3D haptic shading algorithm converges quickly to the correct mean. However, the standard deviations of radius for these models are much higher than those found with the 2D haptic shading algorithm. Both algorithms reach the same level of accuracy around 10° angle difference between adjacent polygons. However, the standard deviation remains submillimeter through 25.7°. Submillimeter distances are incredibly difficult to detect with proprioceptive cues. The mean and standard deviations of the angle error between the surface normal vector and the ideal vector for both the 2D and 3D haptic shading algorithms are comparable. For angle differences between adjacent polygons of 25.7° or less these errors and their standard deviations are less than a tenth of a degree and are imperceptible. The results shown in Table 5.1 show the 3D haptic shading algorithm's ability to create smooth curved surfaces matching the shape represented by the polygonal models used.

Table 5.1 Mean and standard deviation of radius and angle error of normal vectors for a variety of angles between adjacent polygons evaluated using the 3D haptic shading algorithm.

Angle between adjacent line segments	Mean rendered radius in mm (ideal = 100)	Standard deviation of rendered radius ($x10^{-4}$mm)	Mean angle error of surface normal in degrees (ideal = 0)	Standard deviation of angle error of surface normal
51.4°	97.16 mm	19411.7	0.5995°	0.38354
36.0°	99.30 mm	3486.2	0.1659°	0.10665
25.7°	99.84 mm	778.6	0.0681°	0.04353
17.1°	99.96 mm	191.0	0.0196°	0.01236
12.9°	99.99 mm	59.3	0.0084°	0.00512
10.3°	100.00 mm	25.2	0.0043°	0.00270
8.6°	100.00 mm	12.9	0.0029°	0.00177
7.3°	100.00 mm	7.3	0.0019°	0.00108
6.5°	100.00 mm	5.4	0.0018°	0.00099
5.7°	100.00 mm	4.6	0.0019°	0.00109

CHAPTER 6

SECOND EXPERIMENT

6.1 Object Shape Recognition

An object shape recognition task was performed to demonstrate the developed 3D haptic shading algorithm and provide further insight into the CLD device's capability to facilitate exploration and shape recognition. Virtual objects were rendered under four experimental conditions comprised of two options. These tests were conducted both with kinesthetic feedback alone and with combined tactile and kinesthetic feedback. Tactile feedback was provided by a one-dimensional contact location display device. Object models were also rendered with and without haptic shading that created smooth curved objects and rounded the edges of flat-sided objects such as cubes. The addition of rounded edges was theorized 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 previously reported by Frisoli et al. which hampered his subjects' ability to identify simple object shapes [7]. Objects containing smooth curved surfaces (cone, cylinder, and sphere) were rendered as high count polygonal representations when haptic shading was not used. Table 6.1 lists each object and the number of polygons used as well as the angle between adjacent faces for both the shaded and unshaded models. Note that for the models used with shading that the number of polygons is increased by the extra polygons

Table 6.1 Model sizes and angle differences between adjacent polygons (excluding edges) of each object for the shaded and unshaded models.

	Unshaded Models		Shaded Models	
	# of polygons in the model	Angle difference between adjacent polygons (excluding edges)	# of polygons in the model	Angle difference between adjacent polygons (excluding edges)
Cone	130	5.54°	288	10.0°
Cylinder	260	5.54°	288	10.0°
Cube	12	0	44	0
Sphere	4030	5.54°	1188	10.0°
Tetrahedron	4	0	20	0
Extruded Octagon	32	0	96	0
Extruded Triangle	12	0	36	0

needed to create round edges. The round curved objects (cone, cylinder, and sphere) are designed to have sixty-five sides around their circular cross section in the unshaded models and thirty-six sides around their circular cross section in the shaded models. Since the sphere is curved in two dimensions it requires significantly more polygons to render than the other objects even though it has no corners.

Altered models and the developed shading algorithm were used when rendering rounded edges. These rounded edges were formed by the addition of extra triangles along each edge and are more fully explained in Section 6.1.1. The shading algorithm also smoothed the surfaces of the cone, cylinder, and sphere in this case. Contact location on the virtual finger was calculated over a 14 mm arc length of the 20 mm radius virtual finger model and linearly mapped to be displayed over 14 mm of travel along the length of the participant's finger.

In addition to the four above experimental conditions, subject data pools were further divided into two groups, experienced and inexperienced. Experienced subjects participated in the 2D experiment described in Chapter 4. Inexperienced subjects had no prior experience with the CLD device. Some of the inexperienced subjects had prior haptics experience with other devices.

Subjects were asked to identify the objects rendered from among a list of seven primitive object shapes: cone, cylinder, cube, sphere, tetrahedron, extruded octagon, and extruded triangle (see Figure 6.1). A copy of Figure 6.1 was provided to the participants as a reference. These objects were selected during pilot testing as being distinct, while providing opportunity for confusion between similarly shaped objects, depending on the rendering conditions. Forces were rendered using a Phantom Premium 1.5 while tactile feedback was rendered using the CLD device shown in Figures 2.3, 2.4, and 4.2. The one-dimensional gimbal was modified to allow additional motion from side-to-side but only the tilt angle was monitored. The user's finger was limited to only pointing forward and tilting up and down.

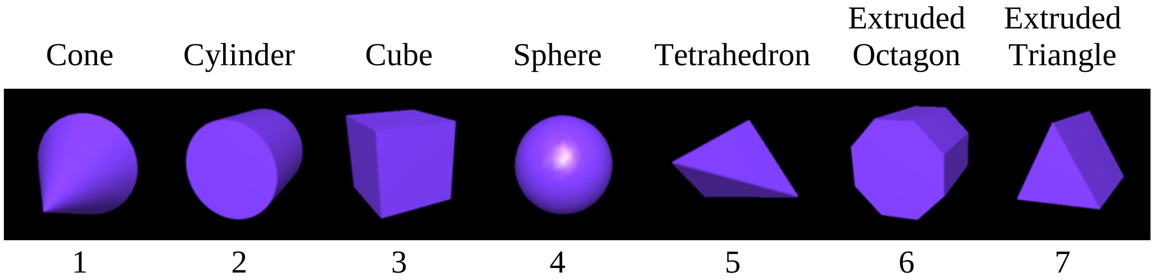


Figure 6.1 The seven models the subjects had to identify. Each object was rendered under four experimental conditions: 1) with and without tactile feedback and 2) with and without rounded edges. Objects were identified by the participants using the numbers under the shapes.

6.1.1 Model Design

The seven models were chosen as a selection of primitive objects containing shapes that could be confused for one another. 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. Participants were not informed of this to prevent exploration strategies involving finding a particular feature but rather instead exploring the model.

Each of the seven models was created such that its principle axis of rotation was in the same direction. The axis of rotation was placed along the "y" axis of the phantom which, when facing forward towards the computer monitor, is horizontal and from the left to right. The cone and tetrahedron models are asymmetric along this axis and provide directional information. These models were rendered facing either direction (pointed to both left and right) during the experiment to eliminate the effects of direction.

A second set of seven models with rounded edges was generated by modifying each of the seven models to include a chamfer piece along each of its edges. For example, the top vertex and bottom edge of the cone was chamfered. The sphere contains no edges and is an exception. The surface normal vectors assigned to the object are dependent on the original surfaces of the model. When the 3D haptic shading algorithm smoothes these chamfered edges the original planar faces remain planar and the added chamfers on each edge become rounded and smooth. The result is shown in Figure 6.2 where the blue curve represents the smooth curved surface generated and the black dashed lines represent the original polygonal model. A 1.5mm radius was chosen for the rounded edges and the chamfers were constructed accordingly.

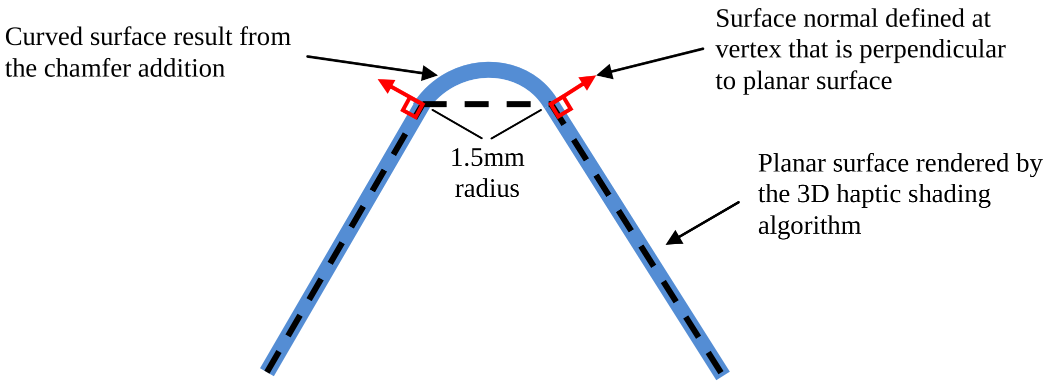


Figure 6.2 A polygonal model cross-section shown with a chamfer piece. The blue curve represents the rendered smooth curved surface while the black dashed lines represent the original polygonal model. The vertex normals are defined by red arrows.

6.2 Experimental Methods and Procedures

Each participant performed a total of four repetitions for each object and rendering condition. These repetitions were broken up into two separate experiment sessions, which were performed with at least a day separation between each day's test. Each day's test session was split into two halves. These halves 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 began without tactile feedback and performed the second half of the experiment with both tactile and kinesthetic feedback. During the portion of the experiment while no tactile feedback was rendered, the CLD device was commanded to a position at the center of the thimble. The roller was then strapped to the thimble and the person's finger to ensure it did not lose contact with the user's finger. Experienced and inexperienced participant groups each contained nine users.

Within each half of the experiment two runs were completed. Each run contains exactly one of each of the seven models rendered with and without rounded edges, providing fourteen total objects to identify. The order in which all fourteen of these objects were rendered was chosen randomly so there was the possibility of receiving the same shape, though not under the same conditions, more than once in succession. Shapes containing directional information (cone and tetrahedron) were rendered facing either direction. During the second run in each half the directions of the cone and tetrahedron were chosen to be opposite of the first run. Thus, in a given half of the experiment, a participant would feel both orientations (left and right pointing) for each object model.

In each trial the participant would explore the currently rendered object and identify it from the list of seven objects to be presented (see Figure 6.1). Once recognized, the participant would press the number key corresponding to the identified shape, e.g., '4' for a sphere. The answer and timing data were recorded and the participant was guided back to the starting position by weak attractive forces (force feedback from the Phantom) and visual feedback of their position. Participants were required to remain at the starting position for one second before continuing. This allows the program time to load the next object model as well as forces the participant to begin each trial at the same relative location above each virtual object. Once the participant had remained at the starting point long enough a "ding" sound was played and visual feedback disappeared to indicate the participant should begin exploring the next object to be identified. The experiment continued until the participant entered answers for all twenty-eight objects presented in the first half. Then a short break was given while the CLD device was adjusted for use in the second feedback condition, that either added or eliminated tactile feedback, depending on the user's starting condition. The remaining twenty-eight trials of the second half were then completed.

Before each half of each day's testing session the participant was given an example surface of an extruded hexagon to help adjust them to the feedback that was being displayed by the CLD device and to help give the participant an understanding of the testing steps. This was done for both the kinesthetic only and for the kinesthetic plus tactile conditions. Visual feedback showing the virtual object and the user's virtual finger on the computer LCD was provided to the user during this pretest. However, no such visual cues were provided while identifying the seven objects during the test, with the exception of the visual cues that guided the user to raise their finger back above the virtual objects after recording their answer for each trial.

The testing apparatus, as shown in Figure 4.2, was identical to the setup used in Chapter 4 during the 2D 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 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 answer as quickly as they felt comfortable. Participants were not permitted to change answers once given.

Seventeen right-handed individuals and one left-handed individual (two females) between the ages of eighteen and thirty-eight participated in the experiments.

6.3 Experiment Results and Discussion

The amount of data collected in the object recognition experiment is too great to present in this chapter. Instead, a summary of important findings is presented. A more comprehensive set of figures and tables is provided in Appendix B. Learning trends were noticed for all participants. A significant difference was found between experienced and inexperienced users in both times and accuracies. Rounding edges was found to increase the ability of participants to follow the contours of objects, but no significant difference was found in accuracies. Rounded objects were found to be both faster and easier to identify. The addition of tactile feedback showed no significant differences in either time or accuracy when identifying objects. Confusion matrices showed the largest confusion was the tetrahedron being identified as a cone. The most unexpected confusion was the extruded triangle being identified as an extruded octagon.

The collected data were examined with respect to 1) shaded vs. unshaded models, 2) models rendered with and without tactile feedback, and 3) experienced vs. inexperienced users. Participants' results involving only correct answers were analyzed first to spot trends. Confusion matrices were analyzed afterward for model correlation.

6.3.1 Learning Trends with Experienced vs. Inexperienced Users

The average performance for the experienced and inexperienced groups can be seen in Figures 6.3 and 6.4. Figure 6.3 shows the number of answers given as the test progresses. Figure 6.4 shows the total errors with respect to the number of answers given. In both figures the second day's experimental results begin where the first day's ends to better demonstrate learning trends. Both of these plots show learning and a significant difference between the experienced and inexperienced groups. Four t-tests were completed to verify the significance of the learning curve and difference between inexperienced and experienced users. From the first experiment session to the second there was a statistical improvement in accuracy ($t(250) = 4.62$, $p < 0.0001$) and time taken to identify objects ($t(1649) = 8.15$, $p < 0.0001$) for both the experienced and

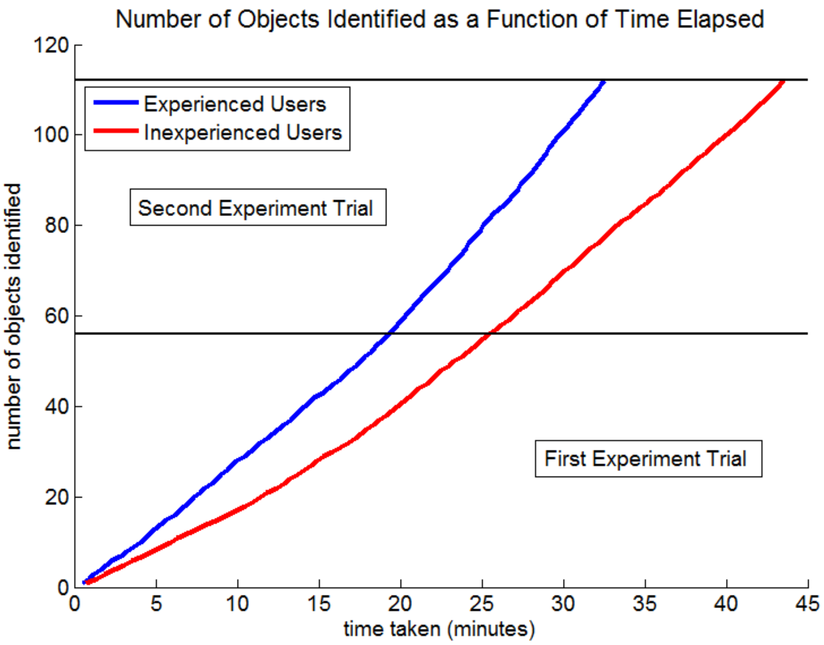


Figure 6.3 Mean total time taken for a number of trials by inexperienced participants (red) and experienced participants (blue). The total time taken was accumulated across both sessions to make learning trends more observable.

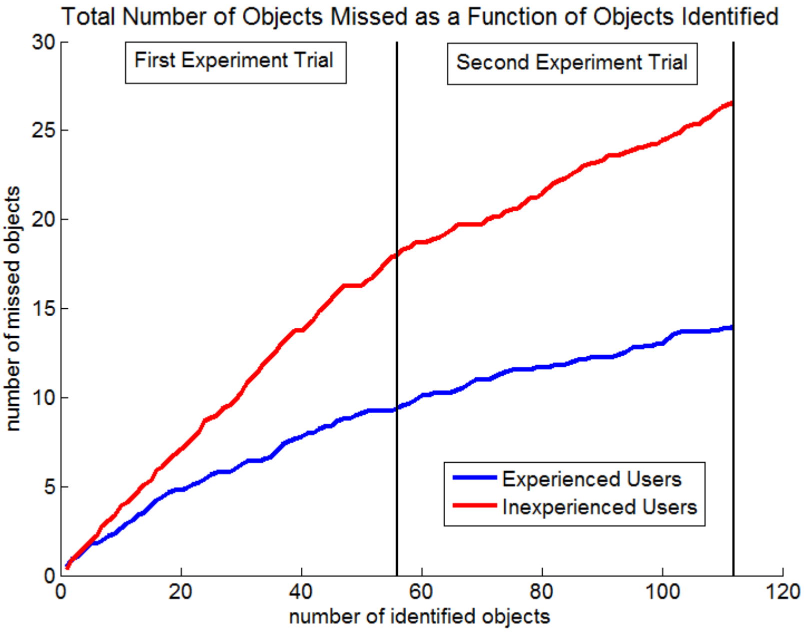


Figure 6.4 Total number of incorrect (missed) answers as a function of trials completed by inexperienced participants (red) and experienced participants (blue). Errors were accumulated across both sessions in order to show learning trends.

inexperienced users together. For both sessions, the experienced users were shown to be significantly more accurate ($t(250) = -4.01$, $p < 0.0001$) and faster ($t(1649) = -5.92$, $p < 0.0001$) than the inexperienced users.

6.3.2 Effects of Rounding Edges

It was found that rounded edges on objects allowed participants to stay in contact with the object's surface for a larger portion of the time spent during exploration ($t(1649) = 37.14$, $p < 0.0001$). This is shown in Figure 6.5. Unexpectedly, there was not a significant difference found in accuracies between objects with and without rounded edges ($t(502) = 1.53$, $p = 0.1277$). Figure 6.6 shows the number of correct answers for all participants for objects with and without rounded edges. As can be seen in Figure 6.6 overall accuracies are relatively high which resulted in a ceiling effect, making it difficult to find statistical differences in performance. That is, the objects were too easily identified to create confusion.

6.3.3 Curved Objects vs. Faceted Objects

Figure 6.7 shows the mean times to answer for all seven objects. The sphere takes nearly half the amount of time to identify as any of the other seven objects ($t(1649) = -10.10$, $p < 0.0001$) and was significantly easier to identify ($t(250) = -4.62$, $p < 0.0001$). This was expected because the sphere contains no corners which makes it unique among the seven objects and easier to identify.

In general participants took less time to identify objects containing round curved surfaces (cone, cylinder, sphere) ($t(1649) = -6.61$, $p < 0.0001$). This is visible in

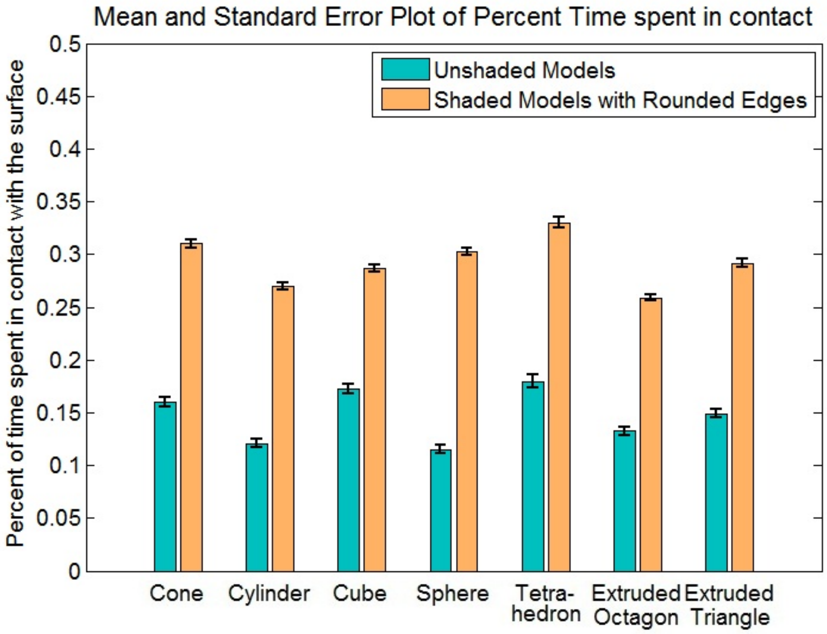


Figure 6.5 Bar plot of the mean and standard error of each of the seven objects and the percent of the **time spent in contact with respect to unshaded models and those with rounded edges**. Participants spent more time in contact with objects containing slightly rounded edges (1.5 mm radius).

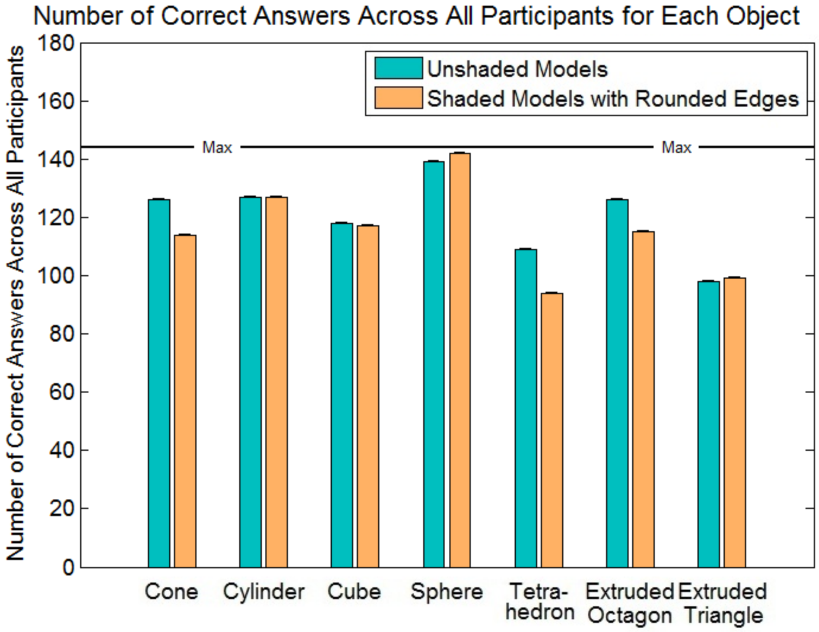


Figure 6.6 Bar plot of each of the **number of correct answers** for seven objects **with respect to models with and without rounded edges**. There is not a statistically significant difference between models with and without rounded edges.

Figure 6.7 where the mean time to answer since the first contact for the cone, cylinder, and sphere is less than for any other shape. The total number of correct answers from all participants is given in Figure 6.8. T-testing verified that round curved objects were not only faster to identify but statistically identified with greater accuracy ($t(250) = -4.91$, $p < 0.0001$). The cone cylinder and sphere have the highest number of correct answers. The cube and extruded octagon both have around the same number of correct answers as the cone. Both the extruded triangle and tetrahedron contain near vertical faces that are difficult for people to interact with when using the current CLD device. This may explain the significantly lower accuracies on these objects.

6.3.4 Participant Exploration Strategies

Independently, participants all seemed to develop the same 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. The statistical difference shown by the recognition time and accuracy of the sphere could be due to this exploration technique since only the side-to-side motions are required to identify the sphere.

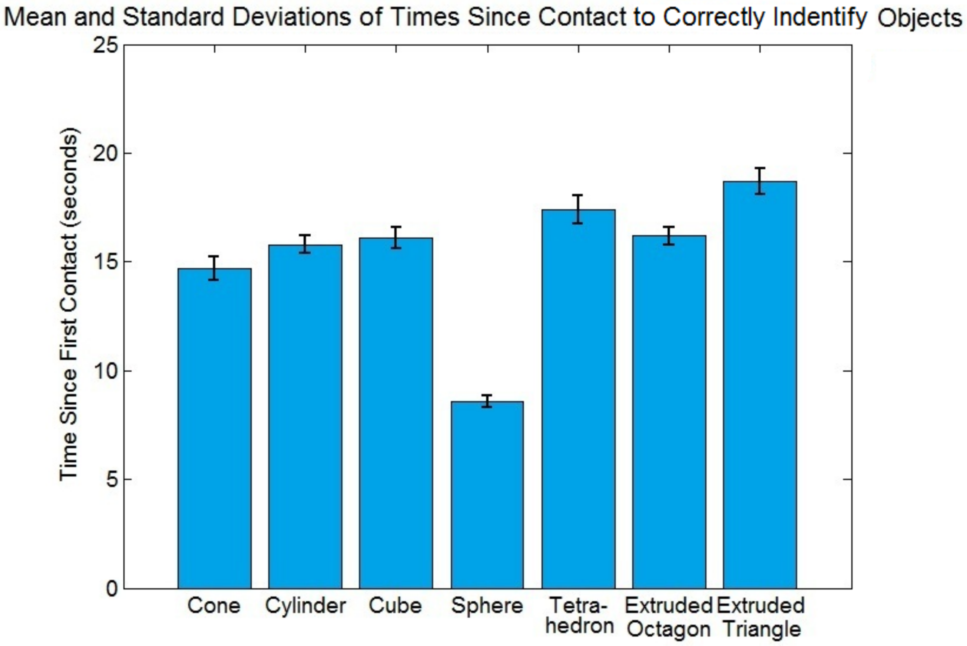


Figure 6.7 Bar plot of the mean and standard error of the **times since initial contact to identify** each of the seven objects. The sphere is almost half the time of all other shapes to identify.

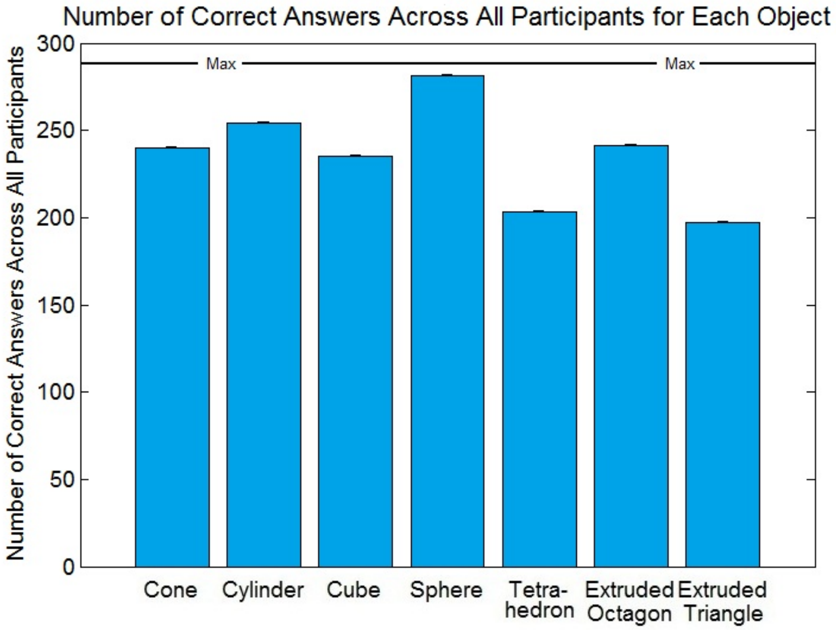


Figure 6.8 Bar plot of each of the **total number of correct answers** given by all participants for each of the seven objects.

6.3.5 Effects of Contact Location Display

User's interactions with the CLD device suggest that the device requires further mechanical revisions before it can provide effective haptic interactions in 3D environments. This was especially noticeable when using the CLD to contact the front face of objects with the tip of the finger. In this situation the dynamics of the device bend the spring steel drive wires away from the user's finger as well as providing a torque on the user's finger that makes the object feel like it is pulling downward (see Figure 6.9). This conflicts with the intended haptic interaction and is distracting. A similar problem exists when interacting with the bottom of the object. Several users independently determined that if their middle finger was used to keep the roller in contact with their finger-pad this issue would be mitigated and the sensations from the device were substantially preferred (see Figure 6.10). This indicates that the next revision of the device should apply kinesthetic feedback through the thimble rather than through the contact element (roller) of the CLD device.

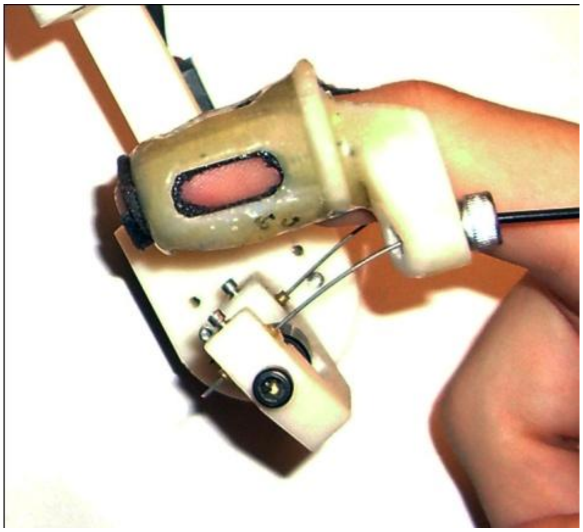


Figure 6.9 The contact location display in contact with a surface using the tip of the finger. The spring steel drive wires bend downward and away from the users finger.

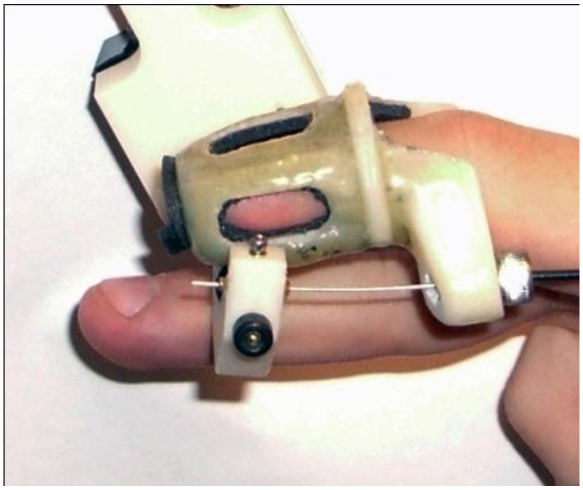


Figure 6.10 A user pressing the contact roller of the contact location display into contact with their finger-pad using their middle finger to increase performance.

The collected data do not seem to indicate any change in identification accuracy (t(502) = 0.544, p = 0.587) or identification time (t(1649) = 0.846, p = 0.398) when using the CLD device (with or without tactile feedback). These conditions are shown in Figures 6.11 and 6.12. This indicates there is no benefit given by the addition of contact location feedback. This, again, could be due to the CLD's current design limitations. Without tactile feedback the device simply became a kinesthetic display. Participants never had problems utilizing the device in this configuration. When the CLD device was used to provide tactile feedback it posed cumbersome restrictions on how it could be used, requiring the user to mostly touch objects from the top side. This implies that whatever benefit the device could have been giving was degraded by these limitations. This theory is further supported when comparing the confusion matrices presented in Section 6.3.6. After redesigning the device to make it more effective within 3D environments there may be a more noticeable improvement in user capabilities to identify objects rendered with contact location feedback.

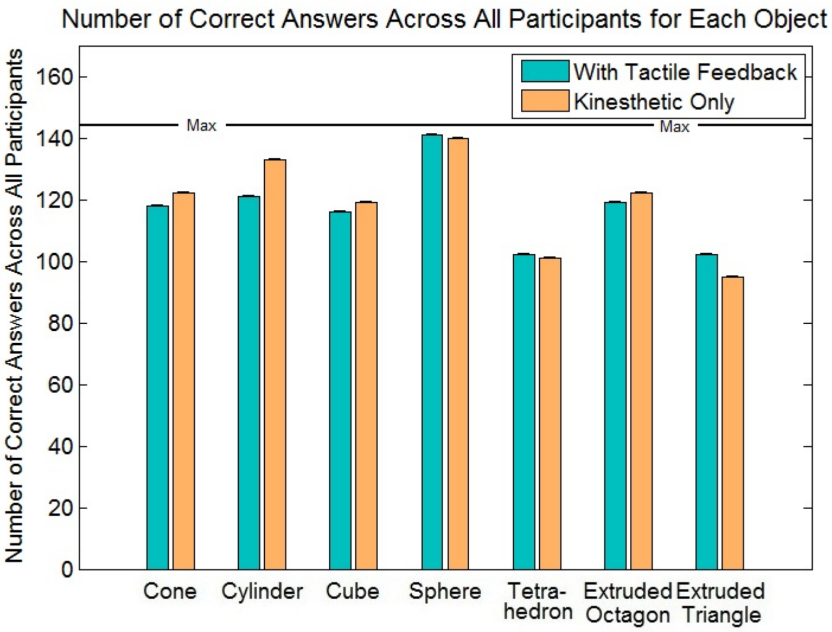


Figure 6.11 Bar plots for each of the seven objects showing the **number of correct responses** among all participants for the cases **with and without tactile display**.

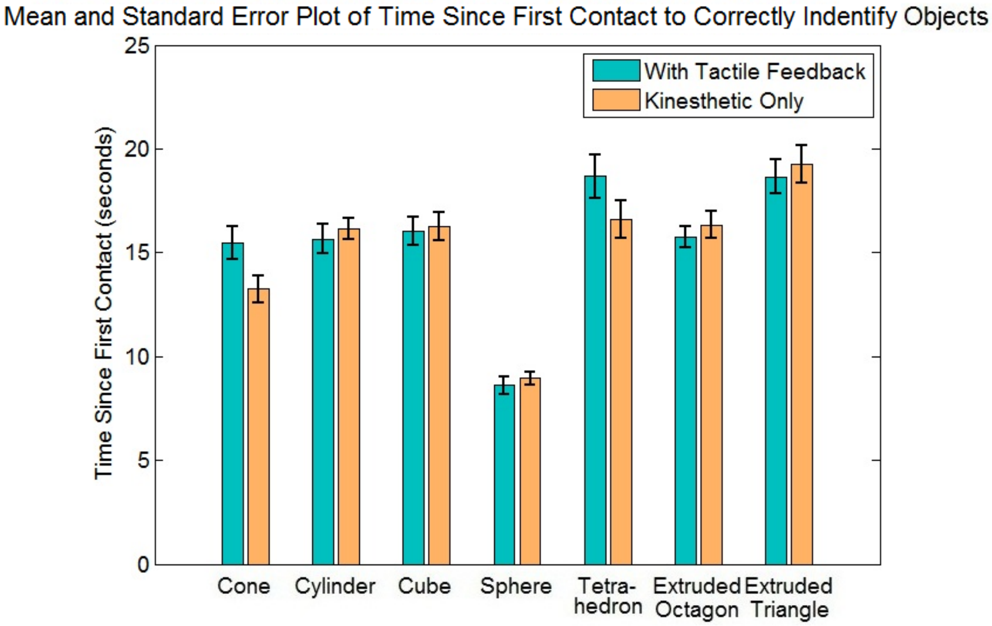


Figure 6.12 Bar plots of the mean and standard error of the seven objects showing the amount of **time to identify** the object since it was first contacted for cases **with and without tactile display**.

6.3.6 Analysis of the Confusion Matrices

The confusion matrix presented in Table 6.2 contains the answers from all participants. The columns represent the shape presented to the participant while the rows represent their responses/answers. The diagonal containing correct answers has been highlighted in grey. Major confusion elements of each column have been highlighted according to their relative strengths with respect to the total number of answers and the mean of the error terms. The confusion matrix shown in Table 6.2 is asymmetric. This implies that participants perceived some objects as others but not vice versa. The most predominant confusion was participants identifying the tetrahedron as a cone and to a lesser extent the cone as a tetrahedron. Identifying the tetrahedron as a cone occurred in a little over 20% of the samples. Weaker, (< 10%) but still predominant, confusions were also observed. Participants confused the extruded octagon with the cylinder but not the cylinder with the extruded octagon. The extruded triangle was confused for the tetrahedron, 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 being identified as the extruded octagon came as a surprise. The extruded triangle's faces are near vertical which is more difficult to interact with. Participants may have been identifying the shape as an extruded octagon by these faces alone rather than fully exploring the model. To the least extent there were small confusions involving the cone identified as a cylinder, the cylinder identified as a cube, and the cube identified as an extruded octagon and extruded triangle. These confusion elements stand out among the other terms in their columns but are weak when compared to the total number of answers (< 6%).

Table 6.2 Confusion matrix of **answers** from **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	240	5	6	1	58	3	7
	Cylinder	13	254	9	2	1	20	12
	Cube	1	17	235	0	1	10	11
	Sphere	5	0	6	281	11	0	2
	Tetrahedron	24	3	2	1	203	5	29
	Extruded Octagon	1	5	14	3	3	241	30
	Extruded Triangle	4	4	16	0	11	9	197

The confusion matrix was then split between unshaded models and shaded models (those with rounded edges). These two matrices are shown below in Tables 6.3 and 6.4 and follow the same coloring scheme. The most predominant confusion in both of these matrices is still the tetrahedron identified as a cone. This confusion is almost evenly split among the two cases with only slightly more confusion when the tetrahedron is rendered with rounded edges. Further investigation into the root of this confusion showed it is independent of all parameters with a slight bias toward models with rounded corners (see Appendix B and Tables 6.3 through 6.8). This also applies to the weaker confusion of the cone as a tetrahedron. The extruded octagon was predominantly confused for the cylinder when its edges were rounded. The extruded triangle was confused for the tetrahedron almost exclusively in the unshaded models. Whether its edges were rounded or not the extruded triangle was capable of being confused with an extruded octagon. The

Table 6.3 Confusion matrix for **unshaded models** for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Unshaded Shape Presented to Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	126	2	2	0	25	1	5
	Cylinder	4	127	3	1	1	4	2
	Cube	0	10	118	0	0	6	6
	Sphere	3	0	2	139	2	0	1
	Tetrahedron	9	1	1	1	109	4	18
	Extruded Octagon	0	3	8	3	1	126	14
	Extruded Triangle	2	1	10	0	6	3	98

Table 6.4 Confusion matrix for **shaded models** containing rounded edges for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shaded Shape Presented to Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	114	3	4	1	33	2	2
	Cylinder	9	127	6	1	0	16	10
	Cube	1	7	117	0	1	4	5
	Sphere	2	0	4	142	9	0	1
	Tetrahedron	15	2	1	0	94	1	11
	Extruded Octagon	1	2	6	0	2	115	16
	Extruded Triangle	2	3	6	0	5	6	99

Table 6.5 Confusion matrix for objects rendered **with contact location feedback** for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Participant with Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	118	4	2	1	23	1	6
	Cylinder	6	121	6	1	1	8	3
	Cube	1	11	116	0	0	7	6
	Sphere	4	0	6	141	8	0	2
	Tetrahedron	13	2	0	0	102	4	14
	Extruded Octagon	0	3	8	1	2	119	11
	Extruded Triangle	2	3	6	0	8	5	102

Table 6.6 Confusion matrix for objects rendered **without contact location feedback** using **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Participant without Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	122	1	4	0	35	2	1
	Cylinder	7	133	3	1	0	12	9
	Cube	0	6	119	0	1	3	5
	Sphere	1	0	0	140	3	0	0
	Tetrahedron	11	1	2	0	101	1	15
	Extruded Octagon	1	2	6	2	1	122	19
	Extruded Triangle	2	1	10	0	3	4	95

Table 6.7 Confusion matrix for **answers** from **experienced users**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Experienced Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	130	0	0	0	19	0	5
	Cylinder	3	136	5	0	0	6	6
	Cube	0	4	122	0	0	1	3
	Sphere	0	0	2	143	7	0	1
	Tetrahedron	11	1	2	0	117	0	14
	Extruded Octagon	0	2	4	1	0	134	15
	Extruded Triangle	0	1	9	0	1	3	100

Table 6.8 Confusion matrix for **answers** from **inexperienced users**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Inexperienced Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	110	5	6	1	39	3	2
	Cylinder	10	118	4	2	1	14	6
	Cube	1	13	113	0	1	9	8
	Sphere	5	0	4	138	4	0	1
	Tetrahedron	13	2	0	1	86	5	15
	Extruded Octagon	1	3	10	2	3	107	15
	Extruded Triangle	4	3	7	0	10	6	97

remaining weakest confusions diverge a little at this point. The cylinder identified as a cube occurs just as often with and without rounded edges on the ends of the cylinder. However, the confusion of the cube with both extruded shapes occurred predominantly in the unshaded case. Users likely had a difficult time following the contours of the cube while it was unshaded. The addition of rounded edges allowed the user to stay in contact with the surface and follow its contours better (see Figure 6.6).

Confusion between objects rendered with and without tactile feedback from the contact location display was also investigated (see Tables 6.5 and 6.6). It is interesting to note that while there is no statistical difference between the number of correct answers given with and without tactile feedback (see Figure 6.12), the majority of confusion cells identified earlier in Table 6.2 are more pronounced without tactile feedback. This supports the theory that the CLD device would decrease confusion by providing additional feedback. This benefit was unfortunately hindered by the limitations on the CLD device and by the low difficulty of the identification experiment.

Finally, the confusion matrix difference between experienced and inexperienced users was evaluated (see Tables 6.7 and 6.8). Other than making fewer overall errors there are some differences between the experienced and inexperienced groups worth noting. Experienced participants rarely made the mistake of identifying a cone for a cylinder, a cylinder for a cube, and a cube for an extruded octagon. Both experienced and inexperienced participants had the same level of trouble identifying the extruded triangle.

6.4 Experiment Conclusions

The addition of rounded edges was shown to significantly improve the time spent in contact with models. Frisoli et al. previously reported that loss of contact with objects hampered his subjects' ability to identify simple object shapes [7]. Our experiments also show that users are more capable of identifying the smooth curved objects (cone, cylinder, sphere) more quickly than faceted objects, with the sphere being the fastest and easiest to identify. The 3D haptic shading algorithm is reconfirmed through its use to properly render the rounded edges.

Participants each developed their own exploratory procedures independently but all came up with roughly the same strategy. Participant's interactions with the CLD device suggest the device requires a redesign before becoming effective in 3D environments, due to the roller deflecting away from the user's fingertip during haptic exploration with the front and bottom of virtual objects. Several participants independently determined that using their other fingers to hold the CLD roller in contact with their finger-pad allowed better interactions with the device. Thus it may be advisable to decouple the tactile and kinesthetic interactions in future designs. Accuracies of participants with and without tactile feedback did not show a statistical difference but there was less specific confusion when tactile feedback was provided.

Confusion matrices were investigated between each of the primary cases: 1) objects with and without rounded corners, 2) objects rendered with and without tactile feedback, and 3) experienced vs. inexperienced users. The strongest confusion occurred with the tetrahedron identified as a cone. This confusion was evaluated to be independent of all parameters.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

Haptic devices are becoming more prevalent throughout the world. Despite this, these devices are still far from their full potential as haptic interfaces. Continuing research into improving these devices is imperative. New devices are arising that are capable of displaying both tactile and kinesthetic information. One such device in particular used in this thesis is the contact location display (CLD). This research emphasizes the development of haptic shading algorithms necessary to fully utilize not only the CLD device but all combined tactile and kinesthetic devices with polygonal model environments.

The contributions of this work are presented in this chapter. A summary of the two shading algorithms as well as the experiments utilizing them is provided. Future work and the potential of this research is discussed.

7.1 Contributions

Three major contributions to the field of haptics were made through this thesis research.

1. 2D haptic shading algorithm. A haptic shading algorithm was developed for two-dimensional polygonal models. This algorithm creates a perfectly smooth curved surface from the underlying polygonal model using Bézier curves.

Since the algorithm makes the facets and edges imperceptible, a significant reduction in model size of complex models is possible.

2. Perception thresholds for smoothly rendered polygonal models. Perception thresholds for angle difference between adjacent polygons were evaluated 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 to ~3°. The full shading algorithm was found to reduce this further allowing up to ~15° angle difference between adjacent polygons before the deviation became noticeable.

3. 3D haptic shading algorithm. An expansion to the 2D haptic shading algorithm was developed. This algorithm creates a perfectly smooth curved surface from the underlying 3D polygonal model using PN triangles. Numerical methods were used to track the surface contact position. Again, the smooth surface makes facets and edges imperceptible allowing for significant reduction in model size of complex models.

7.2 Future Work

The results from this research allow the use of the contact location display with 3D polygonal environments. While this opens numerous other directions of research, the CLD device needs to be revised before continuing. Results from the second experiment involving object recognition indicated that the current contact location display device is ill-suited for interacting with 3D models. The revised CLD device should decouple force feedback from the tactile element to alleviate some of its current issues.

With a redesigned device, research into grasping and manipulation tasks becomes possible. These tasks would be carried out utilizing two contact location displays attached to a users thumb and forefinger. Subjects could be required to pick up and orient a specific object. The addition of contact location feedback should increase both speed and accuracy in this task. Additionally, users could be asked to perform another gross object identification study. With the new ability to partially envelope the object between their two fingers, the object models presented will need to be somewhat complex.

Before the device is redesigned experiments are limited but not fruitless. One proposed experiment for use with the current CLD device evaluates the devices capabilities to assist in feature detection. A series of smooth surfaces would be rendered for the participant each with a small feature somewhere on their surface. It would be the participant's task to locate this feature and identify it. The surfaces would need to be restricted to near horizontal to accommodate the device and provide the highest fidelity interactions.

While the shading algorithms are complete and work well there is still room for improvement. For example, the current 3D haptic shading algorithm utilizes numerical methods to track the surface tangent. By definition numerical methods are unstable given the wrong conditions. Developing a new projection or reparameterization method for guaranteed stability is an area of future research that would be worthy of investigation.

Each of the haptic shading algorithms is capable of smoothing a dynamically updating surface. Thus another avenue of investigation opened by this research is into low resolution deformable meshes. Evaluating this research may prove difficult but there are many benefits to providing easily manipulated meshes.

The work done in this thesis presents a method of shading the edges of objects to improve one's ability to maintain contact with a surface. Further research into the optimal radius of rounded edges, so that they are still perceived as an edge while increasing the ability of users to follow contours, is currently being investigated with our collaborators in Prof. Hong Tan's laboratory at Purdue University.

7.3 Conclusions

This thesis describes two developed haptic shading algorithms for polygonal models. These algorithms produce smooth curved surfaces that are superior to conventional shading techniques in that they are applicable for both haptic and tactile rendering. All devices capable of being rendered using a proxy based method are capable of using these algorithms. They can serve as replacements to Morganbesser and Srinivasan's shading algorithm [23]. Perception data of prior rendering and shading techniques as well as with the CLD device and the new shading algorithm were collected. These results can be used as a guide in constructing haptic models or reducing complexity in existing models. An object shape recognition task was presented as a means of demonstrating the 3D shading algorithm and to provide further insight into the CLD device's capability to facilitate dexterous exploration and shape recognition. The results from this task suggest that further research and design are needed to fully utilize the CLD device in 3D environments. The work presented in this thesis is integral to advancing haptic devices to extend user interaction beyond audio and visual responses and into a touchable virtual 3D world.

APPENDIX A

PSEUDO CODE FOR IMPLEMENTING THE ALGORITHMS

The next two sections of pseudo code demonstrate a simple implementation of the algorithms presented in Chapters 3 and 5. The first section adds the 2D algorithm to a simple proxy based system. The second section repeats this process for the 3D algorithm.

A.1 Pseudo code for the 2D algorithm

```
compute finger projection to the current tangent line
use the contact point of this projection compute the angular_fraction
check if angular_fraction is out of bounds
while(angular_fraction out of bounds){
        assign new closest vertex
        compute angular_fraction again
        check if angular_fraction is out of bounds
}
compute t parameter from the angular_fraction
use Bézier curves to compute new tangent line
use proxy methods to compute interactions with the tangent line
repeat haptic loop
```

A.2 Pseudo code for the 3D algorithm

```
compute finger projection to the current tangent plane
distance = length of (current_position - projected_position)
error_vector = new_contact_point - previous_contact_point
old_uvw_point = barycentric coordinates of the old_tracked_point
new_uvw_point = barycentric coordinates of the (old_tracked_point + error_vector)
compute curvature based on old_uvw_point and new_uvw_point at the old_uvw_point
gain = 1/(1+Gd*abs(distance))/(1+Gk*curvature)
while(length of the error_vector >= 1e-5){
        new_tracked_point = old_tracked_point + gain*error_vector
        compute the barycentric coordinates of the new_tracked_point
        use PN triangles equations to define a new tangent plane
        compute finger projection to the current tangent plane
        distance = length of (current_position - projected_position)
        error_vector = new_contact_point - previous_contact_point
        old_uvw_point = barycentric coordinates of the old_tracked_point
        new_uvw_point = barycentric coordinates of (old_tracked_point + error_vector)
        compute curvature based on old/new_uvw_point at the old_uvw_point
        gain = 1/(1+distance)/(1+curvature)
}
use proxy methods to compute interactions with the final used tangent plane
repeat haptic loop
```

APPENDIX B

OBJECT SHAPE RECOGNITION RESULTS

The next two plots, Figures B.1 and B.2, show the effects of learning and the differences between experience and inexperienced users. Figure B.1 shows the number of answers given as time of the test progresses. Figure B.2 shows the number of missed answers as the number of answers given increases. In both figures the second day's experimental results begin where the first day's ends to better demonstrate learning trends.

Figures B.3 and B.4, show the combined data for all users in all cases. Figure B.3 shows the total number of correct answers for each of the seven objects. Figure B.4 shows the mean and standard errors of the time since the object was first contacted to answer for each object.

The subsequent plots, Figures B.5 and B.6, compare the effects of rounding edges on the models. Figure B.5 shows the number of correct answers in each case for each object. Figure B.6 shows the percent of time spent in contact with the surface of the model since it was first contacted in each case for each object.

Figures B.7 and B.8 show the effects of adding tactile feedback. As with the other sets of figures, Figure B.7 shows the number of correct answers from all participants split between the two cases for all objects. Figure B.8 shows the mean and standard error of the amount of time taken since first contact to answer in the two cases for all objects.

Figures B.9 and B.10 further split the collected data into four subgroups comprised of whether or not tactile feedback was provided and whether or not the edges were rounded. Figure B.9 shows the number of correct answers in each of these cases for all objects. Figure B.10 shows the mean and standard error of the time since first contact to answering in these four cases for each of the seven objects.

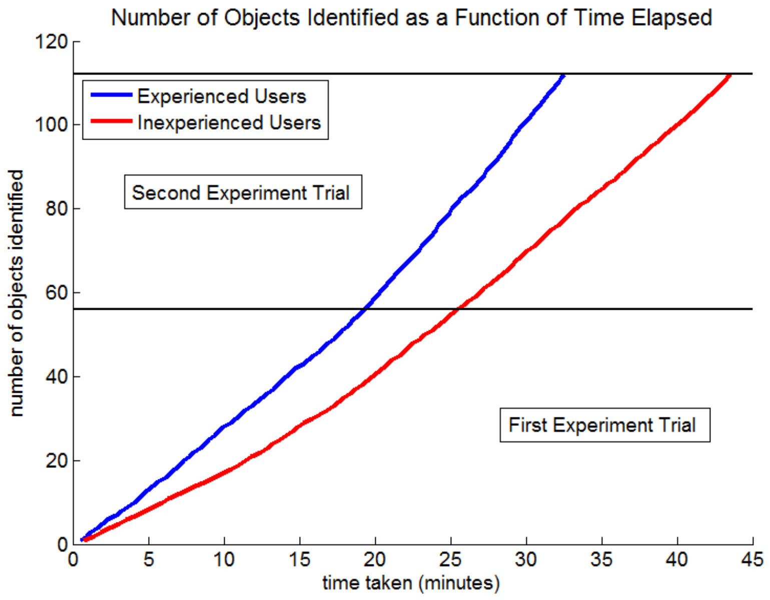


Figure B.1 Mean total time taken for a number of trials by inexperienced participants (red) and experienced participants (blue). The total time taken was accumulated across both sessions to make learning trends more observable.

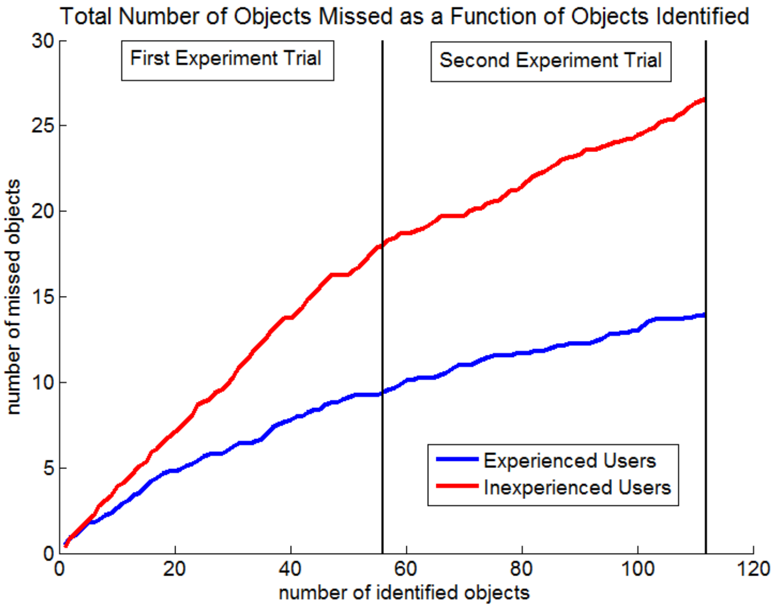


Figure B.2 Total number of missed answers as a function of trials completed by inexperienced participants (red) and experienced participants (blue). Errors were accumulated across both sessions in order to show learning trends.

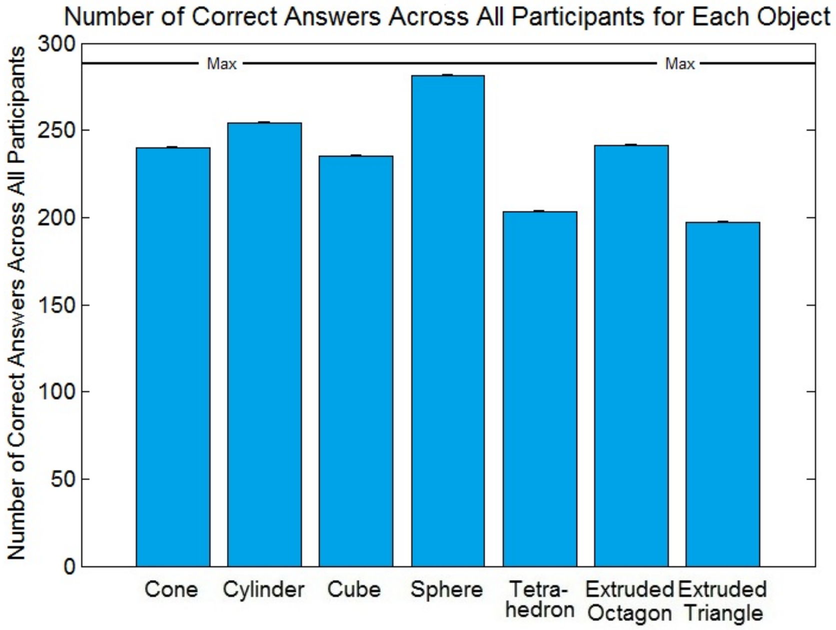


Figure B.3 Bar plot of each of the **total correct responses** given by all participants for each of the seven objects.

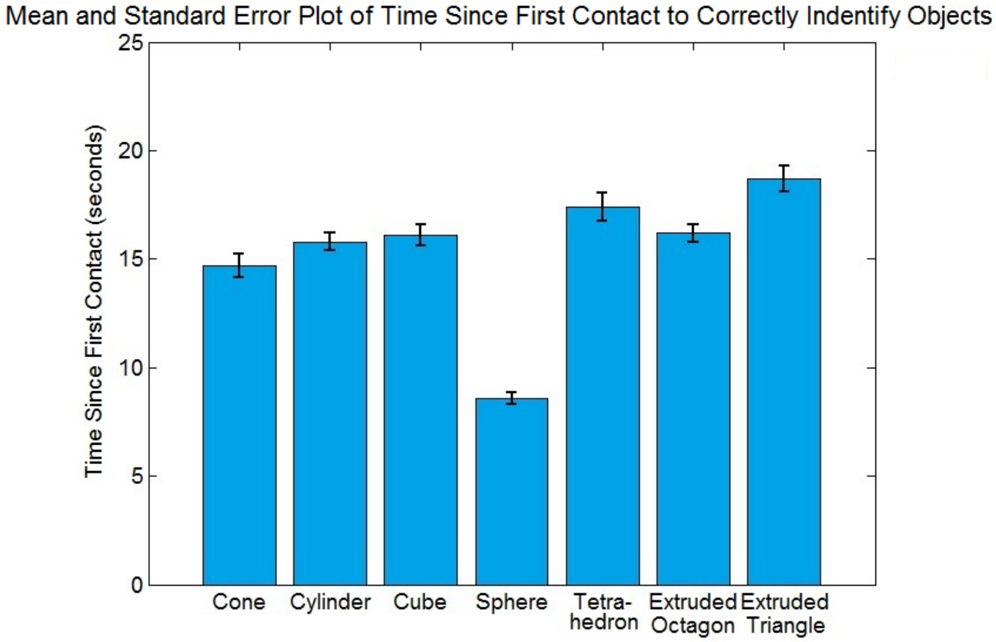


Figure B.4 Bar plot of the mean and standard error of the **times since initial contact to identify each of the seven objects**. The sphere is almost half the time of all other shapes to identify.

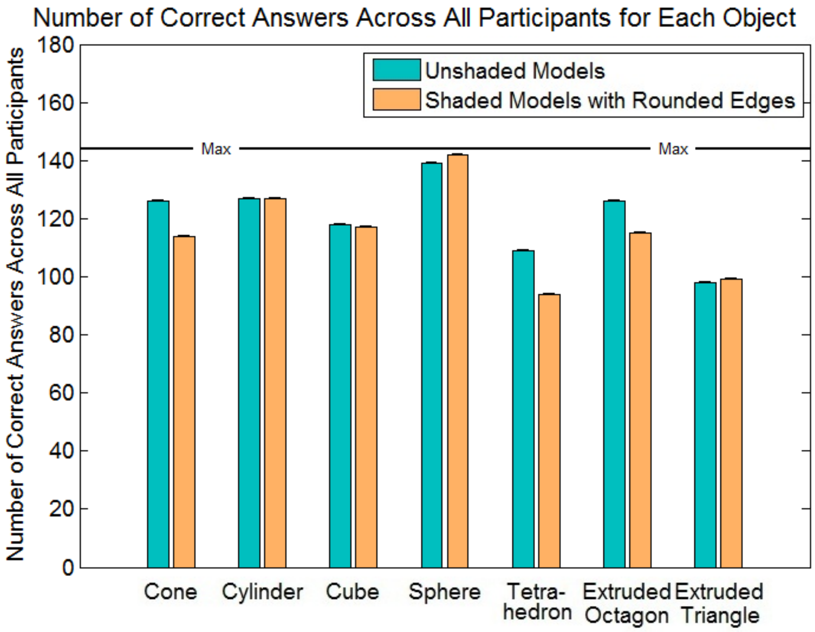


Figure B.5 Bar plot of each of the **<u>number of correct answers</u>** for seven objects **<u>with respect to models with and without shading (rounded edges)</u>**. There is not a statistically significant difference between models with and without rounded edges.

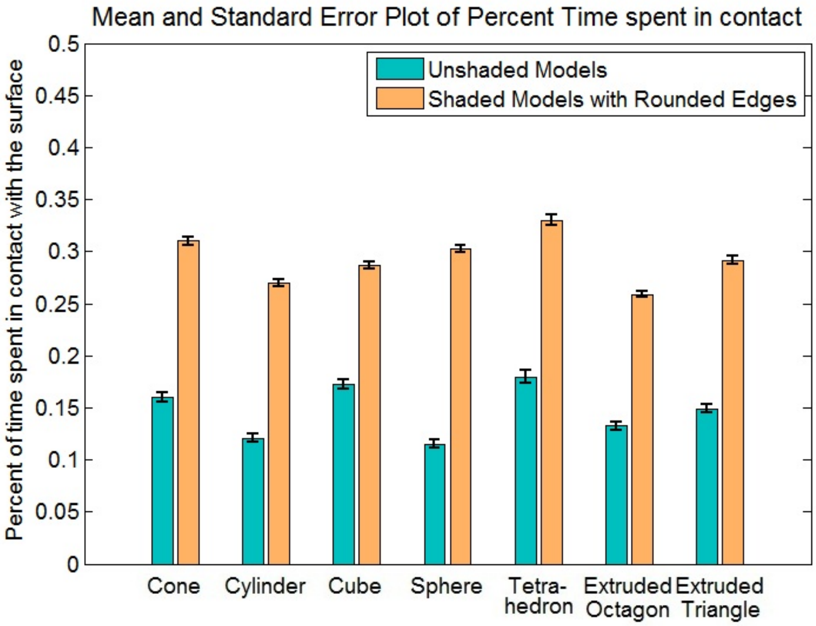


Figure B.6 Bar plot of the mean and standard error of each of the seven objects and the **<u>percent of the time spent in contact with respect to unshaded models and those with rounded edges</u>**. Participants spent more time in contact with objects containing slightly rounded edges (1.5 mm radius).

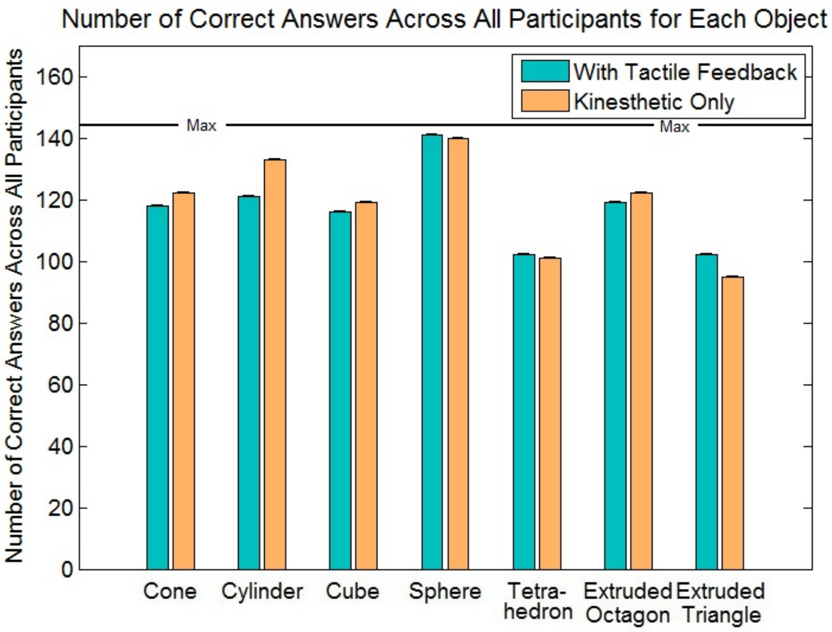


Figure B.7 Bar plots for each of the seven objects showing the **number of correct responses** among all participants for the cases **with and without tactile display**.

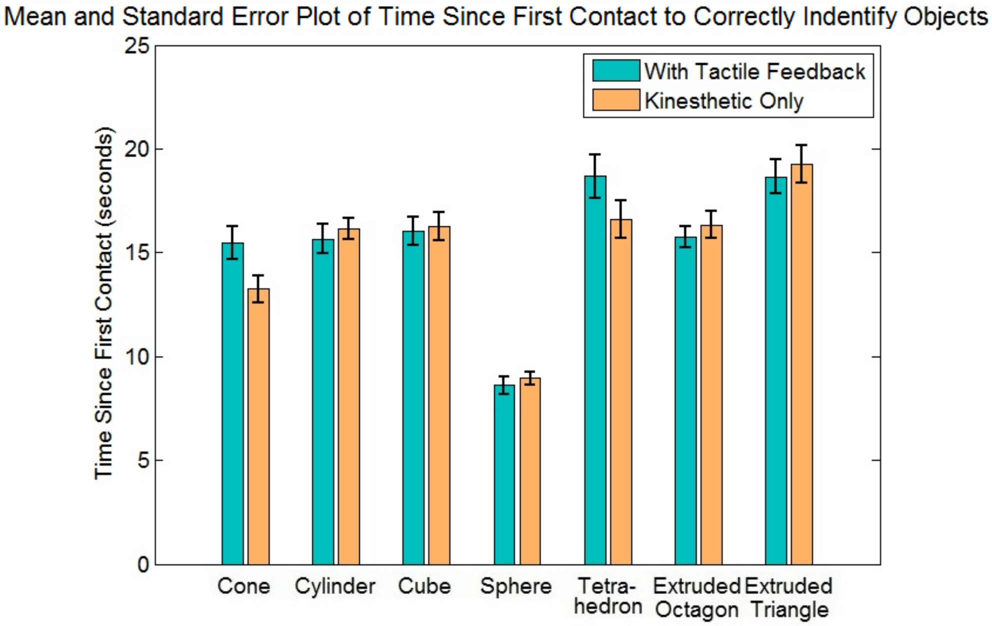


Figure B.8 Bar plots of the mean and standard error of the seven objects showing the **amount of time to identify** the object since it was first contacted for cases **with and without tactile display**.

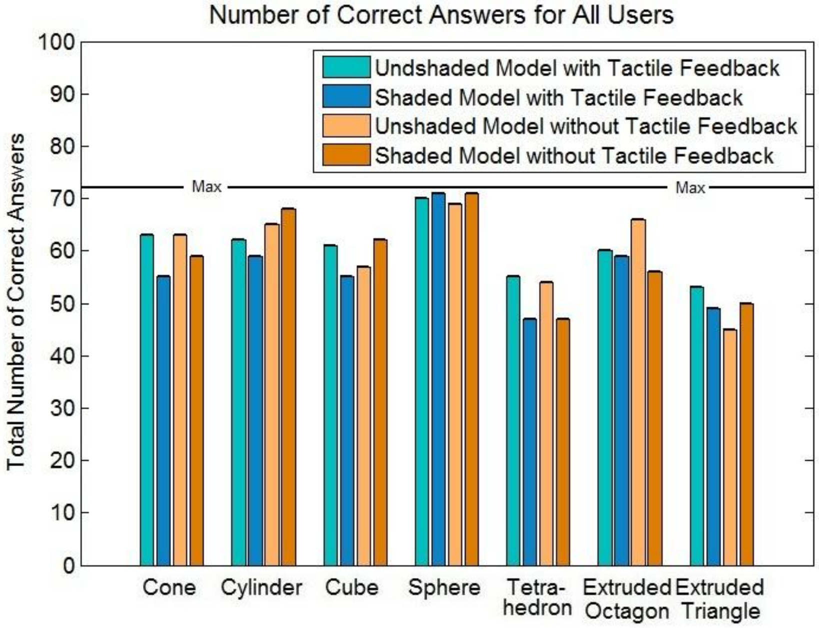


Figure B.9 Bar plots for each of the seven objects showing the **number of correct responses** for **all four cases investigated**.

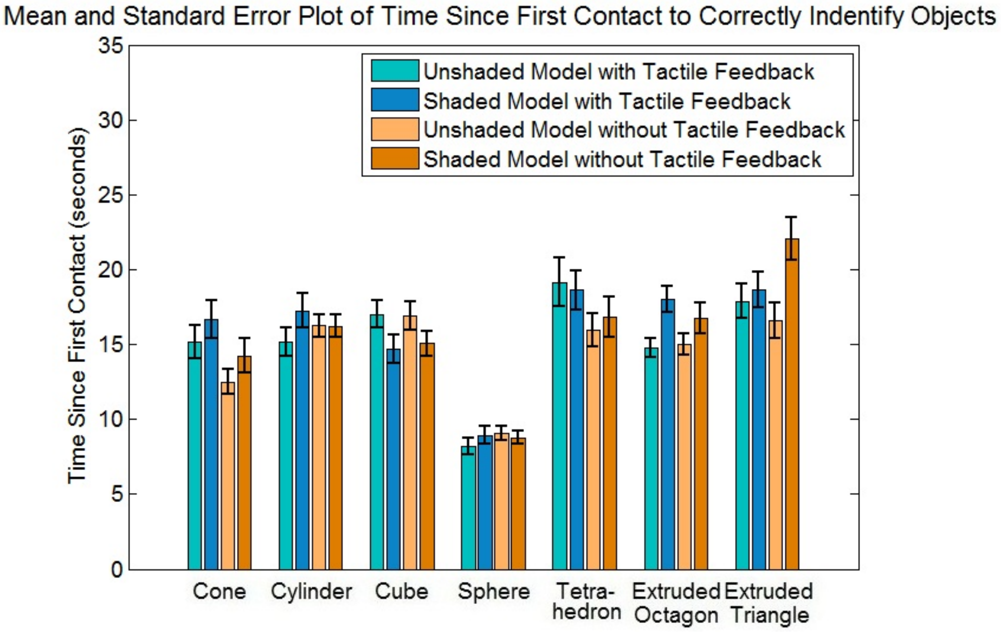


Figure B.10 Bar plots of the mean and standard error of the seven objects showing the **amount of time to identify** the object for **all four cases investigated**.

The next two plots, Figures B.11 and B.12, show the differences between experienced and inexperienced participants. Figure B.11 shows the total number of correct answers in each group for each object. Figure B.12 shows the mean and standard error of the time since first contact to answering for experienced and inexperienced users and each object.

Table B.1 is the confusion matrix for all answers from all participants. All confusion elements should show on this table. Each column represents the shape that was given to be identified. Each row represents the answer given by the participants. The diagonal, correct answers, has been highlighted in grey. Confusion elements have been highlighted with intensity depending on the strength of the confusion with respect to the total number of answers and the other error values in the column.

Tables B.2 and B.3 are the confusion matrices for unshaded and shaded models respectively. The sum of these two matrices is the total data shown in Table B.1. Each column represents the shape that was given to be identified. Each row represents the answer given by the participants. The diagonal, correct answers, has been highlighted in grey. Confusion elements have been highlighted with intensity depending on the strength of the confusion with respect to the total number of answers and the other error values in the column.

Tables B.4 and B.5 are the confusion matrices comparing the addition of contact location tactile feedback or not respectively. The sum of these two matrices is the total data shown in Table B.1. Each column represents the shape that was given to be identified. Each row represents the answer given by the participants. The diagonal, correct answers, has been highlighted in grey. Confusion elements have been highlighted

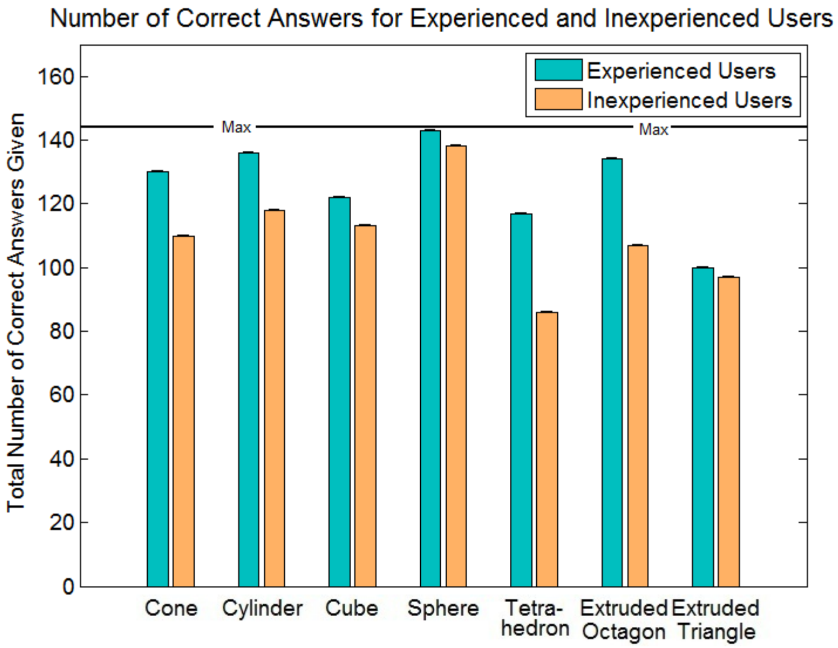


Figure B.11 Bar plots for each of the seven objects showing the **number of correct responses** between **experienced and inexperienced users**.

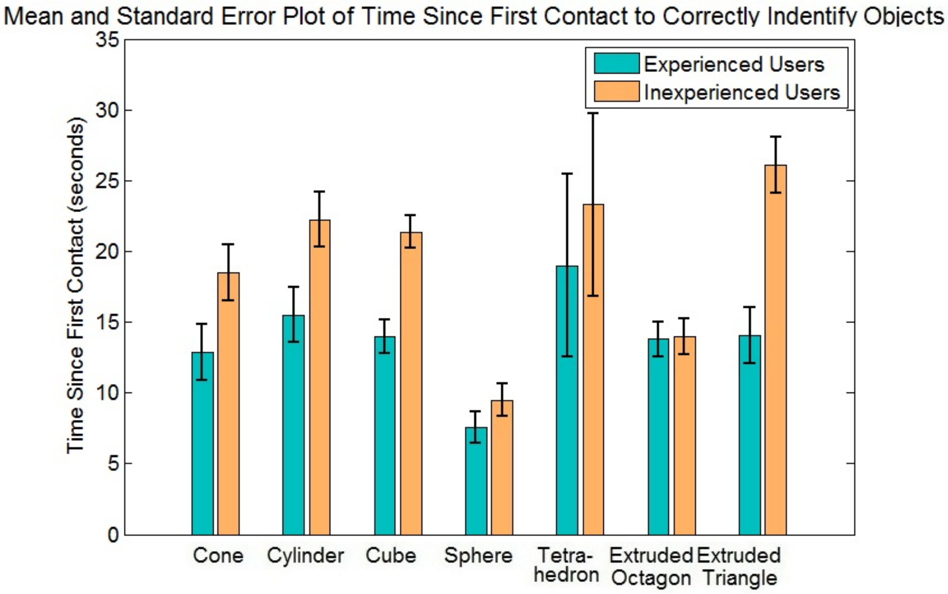


Figure B.12 Bar plots of the mean and standard error of the seven objects showing the **amount of time to identify** the object since it was first contacted for **experienced and inexperienced users**.

Table B.1 Confusion matrix of **answers** from **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	240	5	6	1	58	3	7
	Cylinder	13	254	9	2	1	20	12
	Cube	1	17	235	0	1	10	11
	Sphere	5	0	6	281	11	0	2
	Tetrahedron	24	3	2	1	203	5	29
	Extruded Octagon	1	5	14	3	3	241	30
	Extruded Triangle	4	4	16	0	11	9	197

Table B.2 Confusion matrix for **unshaded models** for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Unshaded Shape Presented to Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	126	2	2	0	25	1	5
	Cylinder	4	127	3	1	1	4	2
	Cube	0	10	118	0	0	6	6
	Sphere	3	0	2	139	2	0	1
	Tetrahedron	9	1	1	1	109	4	18
	Extruded Octagon	0	3	8	3	1	126	14
	Extruded Triangle	2	1	10	0	6	3	98

Table B.3 Confusion matrix for **shaded models** containing rounded edges for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shaded Shape Presented to Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	114	3	4	1	33	2	2
	Cylinder	9	127	6	1	0	16	10
	Cube	1	7	117	0	1	4	5
	Sphere	2	0	4	142	9	0	1
	Tetrahedron	15	2	1	0	94	1	11
	Extruded Octagon	1	2	6	0	2	115	16
	Extruded Triangle	2	3	6	0	5	6	99

Table B.4 Confusion matrix for objects rendered **with contact location feedback** for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Participant with Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	118	4	2	1	23	1	6
	Cylinder	6	121	6	1	1	8	3
	Cube	1	11	116	0	0	7	6
	Sphere	4	0	6	141	8	0	2
	Tetrahedron	13	2	0	0	102	4	14
	Extruded Octagon	0	3	8	1	2	119	11
	Extruded Triangle	2	3	6	0	8	5	102

Table B.5 Confusion matrix for objects rendered **without contact location feedback** using **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Participant without Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	122	1	4	0	35	2	1
	Cylinder	7	133	3	1	0	12	9
	Cube	0	6	119	0	1	3	5
	Sphere	1	0	0	140	3	0	0
	Tetrahedron	11	1	2	0	101	1	15
	Extruded Octagon	1	2	6	2	1	122	19
	Extruded Triangle	2	1	10	0	3	4	95

with intensity depending on the strength of the confusion with respect to the total number of answers and the other error values in the column.

Tables B.6 and B.7 are used to compare the differences in confusion between experienced users and inexperienced users respectively. The sum of these two matrices is the total data shown in Table B.1. Each column represents the shape that was given to be identified. Each row represents the answer given by the participants. The diagonal, correct answers, has been highlighted in grey. Confusion elements have been highlighted with intensity depending on the strength of the confusion with respect to the total number of answers and the other error values in the column.

The next four tables, Tables B.8 through B.11, show confusion matrices broken down by both whether objects were shaded and by whether contact location feedback was provided to the users. These tables help look at all factors simultaneously to see where confusion could be originating. The sum of these four matrices is the total data shown in Table B.1. Each column represents the shape that was given to be identified. Each row represents the answer given by the participants. The diagonal, correct answers, has been highlighted in grey. Confusion elements have been highlighted with intensity depending on the strength of the confusion with respect to the total number of answers and the other error values in the column.

Table B.6 Confusion matrix for **answers** from **experienced users**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Experienced Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	130	0	0	0	19	0	5
	Cylinder	3	136	5	0	0	6	6
	Cube	0	4	122	0	0	1	3
	Sphere	0	0	2	143	7	0	1
	Tetrahedron	11	1	2	0	117	0	14
	Extruded Octagon	0	2	4	1	0	134	15
	Extruded Triangle	0	1	9	0	1	3	100

Table B.7 Confusion matrix for **answers** from **inexperienced users**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shape Presented to Inexperienced Participant						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	110	5	6	1	39	3	2
	Cylinder	10	118	4	2	1	14	6
	Cube	1	13	113	0	1	9	8
	Sphere	5	0	4	138	4	0	1
	Tetrahedron	13	2	0	1	86	5	15
	Extruded Octagon	1	3	10	2	3	107	15
	Extruded Triangle	4	3	7	0	10	6	97

Table B.8 Confusion matrix for **unshaded objects** rendered **with contact location feedback** using **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Unshaded Shape Presented to Participant with Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	63	1	1	0	10	1	4
	Cylinder	1	62	2	1	1	1	0
	Cube	0	5	61	0	0	4	2
	Sphere	3	0	2	70	2	0	1
	Tetrahedron	5	1	0	0	55	4	9
	Extruded Octagon	0	2	4	1	0	60	3
	Extruded Triangle	0	1	2	0	4	2	53

Table B.9 Confusion matrix for **shaded objects** with rounded edges rendered **with contact location feedback** using **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shaded Shape Presented to Participant with Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	55	3	1	1	13	0	2
	Cylinder	5	59	4	0	0	7	3
	Cube	1	6	55	0	0	3	4
	Sphere	1	0	4	71	6	0	1
	Tetrahedron	8	1	0	0	47	0	5
	Extruded Octagon	0	1	4	0	2	59	8
	Extruded Triangle	2	2	4	0	4	3	49

Table B.10 Confusion matrix for **unshaded objects** rendered **without contact location feedback** for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Unshaded Shape Presented to Participant without Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	63	1	1	0	15	0	1
	Cylinder	3	65	1	0	0	3	2
	Cube	0	5	57	0	0	2	4
	Sphere	0	0	0	69	0	0	0
	Tetrahedron	4	0	1	1	54	0	9
	Extruded Octagon	0	1	4	2	1	66	11
	Extruded Triangle	2	0	8	0	2	1	45

Table B.11 Confusion matrix for **shaded objects** with rounded edges rendered **without contact location feedback** for **all participants**. The diagonal has been highlighted in grey. Major confusion values have been highlighted according to their relative strength for each column.

		Shaded Shape Presented to Participant without Tactile Feedback						
		Cone	Cylinder	Cube	Sphere	Tetrahedron	Extruded Octagon	Extruded Triangle
Shape Identified by Participant	Cone	59	0	3	0	20	2	0
	Cylinder	4	68	2	1	0	9	7
	Cube	0	1	62	0	1	1	1
	Sphere	1	0	0	71	3	0	0
	Tetrahedron	7	1	1	0	49	1	6
	Extruded Octagon	1	1	2	0	0	56	8
	Extruded Triangle	0	1	2	0	1	3	50

REFERENCES

[1] Y. Adachi, T. Kumano, and K. Ogino. Intermediate representation for stiff virtual objects. In Virtual Reality Annual International Symposium (VRAIS'95), pages 203–210, 1995.

[2] F. Barbagli, K. Salisbury, C. Ho, C. Spence, and H. Z. Tan. Haptic discrimination of force direction and the influence of visual information. In ACM Transactions on Applied Perception, Vol. 3, No. 2, pp. 125-135, 2006.

[3] E. Cohen, R. Riesenfeld, G. Elber. Geometric modeling with splines: an introduction. Mass: AK Peters, ch. 5-11, 2001.

[4] J. Daniels, C. T. Silva, J. Shepherd, and E. Cohen. Quadrilateral mesh simplification. In ACM SIGGRAPH Asia 2008 Papers, Singapore, December 10-13, 2008.

[5] H. Dostmohamed and V. Hayward. Contact location trajectory on the fingertip as a sufficient requisite for illusory perception of haptic shape and effect of multiple contacts. In F. Barbagli, D. Prattichizzo, and Salisbury, editors, Springer Tracts in Advanced Robotics, vol. 18, pp. 189-198, 2005.

[6] G. Farin. Curves and surfaces for CAGD: A Practical Guide, 3rd ed. Academic Press, 1993.

[7] A. Frisoli, M. Bergamasco, S. Wu, and E. Ruffaldi. 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, 2005.

[8] A. Frisoli, M. Solazzi, F. Salsedo, and M. Bergamasco. A fingertip haptic display for improving curvature discrimination. Presence: Teleoperators and Virtual Environments, 17(6), pp. 550-561, December, 2008.

[9] M. Fritschi, M. Ernst, and M. Buss. Integration of kinesthetic and tactile display – a modular design concept. In 2006 EuroHaptics Conference, 2006.

[10] G. A. Gescheider. Psycophysics: The Fundamentals. 3rd ed: Lawrence Erlbaum Associates, 1997.

[11] H. Iwata and H. Noma. Volume haptization. In IEEE 1993 Symposium on Research Frontiers in Virtual Reality, pages 16–23, 1993.

[12] J. Snyder. An interactive tool for placing curved surfaces without interpretation. In Proceedings of SIGGRAPH 1995, Los Angeles, pp. 209-218, August. 6-11, 1995.

[13] D. Johnson and P. Willemsen. Accelerated haptic rendering of polygonal models through local descent. In Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems (HAPTICS 2004), pages 18–23, March 2004.

[14] D. Johnson and P. Willemsen. Six degree-of-freedom haptic rendering of complex polygonal models. In Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems (HAPTICS 2003), pages 229–235, March 2003.

[15] J.-K. Seong, D. Johnson, G. Elber, and E. Cohen. Critical point analysis using domain lifting for fast geometry queries. Submitted to the Journal of Computer-Aided Design, 2008.

[16] Y. Kim, M. Otaduy, M. Lin, and D. Manocha. Six degree-of-freedom haptic display using localized contact computations. In Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems (HAPTICS 2002), pages 209–216, March 2002.

[17] R. Klatzky and S. Lederman. Touch. In A. F. Healy and R. W. Proctor, editors, Handbook of Psychology, volume 4: Experimental Psychology, chapter 6, pp. 147-176. John Wiley and Sons, 2003.

[18] K. Kuchenbecker, W. Provancher, G. Niemeyer, and M. Cutkosky. Haptic display of contact location. In Proceedings of the IEEE Haptics Symposium, pp. 40-47, 2004.

[19] S. Laycock, A. Day. A survey of haptic rendering techniques. In Computer Graphics Forum 26 (2007), pages 50–65, 2007.

[20] H. Levitt, Transformed up-down methods in psychoacoustics. In Journal of the Acoustical Society of America, vol. 49, pp. 467-477, 1971.

[21] W. McNeely, K. Puterbaugh, and J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1999), pages 401–408, 1999.

[22] S. Michael Ortega and S. Coquillart. A six degree-of-freedom god-object method for haptic display of rigid bodies. In IEEE International Conference on Virtual Reality, 2006.

[23] H. Morganbesser and M. Srinivasan. Force shading for shape perception in haptic virtual environments. In Proceedings of the 5th Annual Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, ASME/IMECE, Atlanta GA, DSC:58, 1996.

[24] D. Nelson, D. Johnson, and E. Cohen. Haptic rendering of surface-to-surface sculpted model interaction. In Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems (HAPTICS 1999), 1999.

[25] B. Phong, Illumination for Computer-Generated Images. Doctoral Thesis. UMI Order Number: AAI7402100, The University of Utah, 1973.

[26] W. Provancher, M. Cutkosky, K. Kuchenbecker, and G. Niemeyer. Contact location display for haptic perception of curvature and object motion. International Journal of Robotics Research, 24(9):691-702, 2005.

[27] D. Ruspini and O. Khatib. Haptic display for human interaction with virtual dynamic environments. Journal of Robotic Systems, 18(12):769-783. 2001.

[28] D. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. In Computer Graphics and Interactive Techniques (SIGGRAPH 1997), pages 345–352, August 1997.

[29] M. Salada, J. Colgate, P. Vishton, and E. Frankel. 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, 2005, pp. 132-137, 2005.

[30] K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles. Haptic rendering: programming touch interaction with virtual objects. In ACM SIGGRAPH Symposium on Interactive 3D Graphics (I3D), pages 123–130, April 1995.

[31] V. Theoktisto, M. Fairen, I. Navazo, E. Monclus. Rendering detailed haptic textures. In Workshop on Virtual Reality Interaction and Physical Simulation, 2005.

[32] T. Thompson II, D. Johnson, and E. Cohen. Direct haptic rendering of sculptured models. In ACM SIGGRAPH Symposium on Interactive 3D Graphics (I3D), pages 167–176, 1997.

[33] A. Vlachos, J. Peters, C. Boyd, J. Mitchell. Curved PN triangles. In Symposium on Interactive 3D Graphics, 159-166, 2001.

[34] R. Webster, T. Murphy, L. Verner, and A. Okamura. A novel two-dimensional tactile slip display: design, kinematics and perceptual experiments. ACM Transactions on Applied Perception, 2(2):150-165, 2005.