Tessellation of NURBS surfaces for use in tactile physics simulations

Pieter Carton

Supervisors: R. Venkatesha Prasad, K. Kroep

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Pieter Carton
Email address of the Student: p.a.carton@student.tudelft.nl
Final project course: CSE3000 Research Project
Thesis committee: R. Venkatesha Prasad, K. Kroep, M. Weinmann

Abstract

Tactile Internet (TI) is a networking paradigm with the goal of allowing for transfer of skill by transmitting haptic feedback. Accurate haptic feedback requires Ultra Low Latency (ULL), which severely limits the distance over which TI can be used. To address the ULL requirement, a local model can be used to generate haptics. For accurate haptic feedback generation, proper representation of smoothly curved objects is needed inside the model. Non Uniform Rational B-Spline (NURBS) surfaces offer an expressive way of representing such objects, but must first be tessellated to a mesh. However, little is known about NURBS tessellation in the context of tactile internet. Therefore, this paper will discuss how to best tessellate NURBS surfaces for use in haptic feedback generation. A global spacing tessellation algorithm was implemented, and the resulting meshes were analyzed. Furthermore, a comprehensive user study was conducted to investigate the mesh quality necessary for smooth objects. Findings imply that careful selection of the tolerance is needed, to avoid over-burdening the model. A mesh tolerance of $\epsilon = 0.05mm$ is proposed as a good balance between spatial fidelity and simplicity of tessellated meshes.

1 Introduction

The Tactile Internet (TI) is an emerging networking paradigm, with the goal of allowing *transfer of skill* over the Internet. Tactile internet achieves this by enabling long-range manipulation of robotic devices by a human operator [1]. Crucially, the operator receives haptic feedback from the robotic device's interactions with its local environment. This sense of touch should make the operator feel as if physically present in the remote environment. The added sensory dimension that haptic feedback offers could allow for remotely performing work that traditionally relies on a sense of touch, such as surgery or machine maintenance [2].

In order for haptic data to feel convincing, a delay of only 1 ms is permissible between performing an action and feeling its effects [1; 2]. In the worst case, excessive delays of haptic feedback can lead to cyber-sickness, with symptoms such as headaches, nausea, and vertigo [3; 4]. The need for low haptic latency in TI is commonly referred to as Ultra Low Latency (ULL) requirement and poses an important obstacle in the field of tactile internet [5]. Unfortunately, due to network speed being bounded by the speed of light, the ULL requirement makes TI over distances in excess of $150\,\mathrm{km}$ difficult.

One way to circumvent network delays and satisfy the ULL requirement lies in Model Mediation Teleoperation (MMT) [6]. The idea of model mediation is to locally simulate the effects a robotic device's movements will have on its environment through a local model. In the case of tactile internet, a tactile physics simulation can be used as a model to instantly generate haptic feedback after a movement is made.

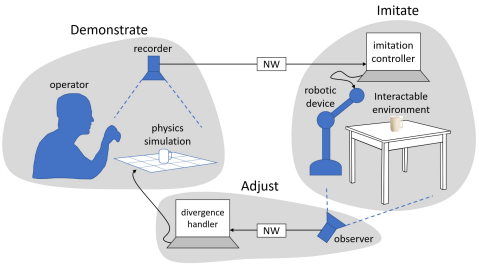


Figure 1: An example of the use of model mediation for tactile internet. First, the operator demonstrates an action inside of a physics simulation, receiving simulated haptic feedback. Then, a robot arm imitates this action in a remote environment. Finally, any divergences in the remote environment are used to adjust the local physics model. Image created by Kees Kroep.

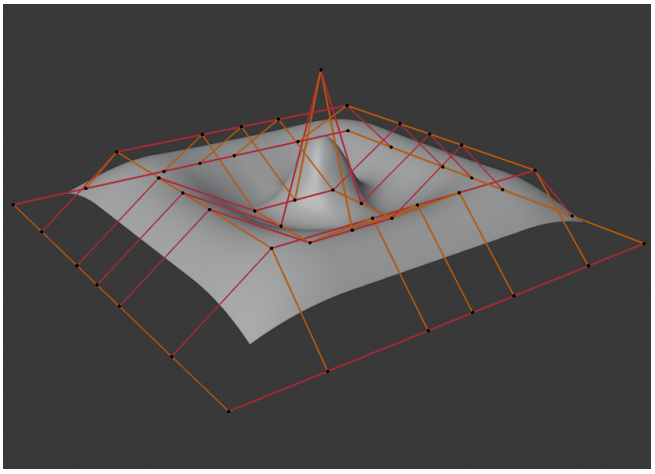


Figure 2: A render of a NURBS surface. The control points dictating the shape of the surface are displayed over the surface. As can be seen, a smooth shape can be created with few control points.

The model parameters, such as the mass, location, and shape of objects surrounding the robotic device, can be estimated using computer vision. This process is illustrated in Fig 1.

The speed and accuracy of the local model are of utmost importance for the quality of the haptic feedback. Therefore, choosing a suitable object representation to use inside the model is vital. Haptics generated from the chosen object representation should mirror the feeling of interacting with its real-life counterpart. As physical objects often feature surfaces with smooth curves, it is imperative that these curves can properly be represented within the object representation. Additionally, the representation should allow for efficient haptic rendering and rigid body dynamics calculations.

Non-Uniform Rational B-Spline (NURBS) surfaces are a promising candidate for the representation of smoothly curved objects. NURBS surfaces are the industry standard in Computer Assisted Design (CAD) for representing complex, organic surfaces [7]. NURBS surfaces are constructed through smooth interpolation between grid control points in three-dimensional space and allow for infinite-resolution

curves. They can therefore represent smooth objects incredibly accurately. An example of such a NURBS surface can be seen in Fig. 2.

Unfortunately, the mathematical complexity of NURBS surfaces makes them unsuitable for direct use in tactile physics simulations. To make the simulation of NURBS surfaces possible, they must therefore first be converted to a mesh through a process called tessellation. However, research on tessellation algorithms has mostly been performed in the context of Computer Assisted Machining (CAM) [8] and computer graphics [9; 10]. In the context of the tactile internet, where touch instead of vision is the leading sense and tight timing constraints are present, little research has been done.

The objective of this paper is to describe a method of tessellating a Non-Uniform Rational Basis-Spline surface for use in tactile physics simulations. Of particular importance are the polygon count and perceived smoothness of meshes generated by the tessellation of NURBS surfaces. After all, these directly correspond to the performance of the model and the accuracy of the mesh. This paper will make the following contributions:

- **NURBS Tessellation Algorithm**: A uniform tessellation algorithm for NURBS was replicated and implemented. Meshes created by this algorithm have an error less than some user-specified tolerance ϵ.
- **Analysis of Tessellated Meshes**: An analysis of the complexity of meshes generated through the implemented uniform tessellation algorithm is performed
- **Mesh Smoothness User Study**: A comprehensive user study was conducted on the mesh tolerance necessary for objects to be perceived as smooth. Participants were surveyed with a haptic controller and tactile physics engine.

This paper is structured as follows. To start off, section 2 briefly discusses related works within the subject of NURBS surface tessellation. Following this, section 3 explains the methodology used to conduct research. Section 4 gives an account of the experimental setup used to obtain results. Subsequently, section 5 provides an analysis of the meshes generated through the implemented algorithms and the outcome of the smoothness perception user study. Next, section 6 summarizes the reproducibility of the obtained results and the use of human participants. Following this, section 7 places obtained results by comparing them to existing literature. Afterwards, recommendations for future avenues of research are discussed in section 8. Finally, section 9 will conclude the findings of this paper.

2 Related Works

The following section will give an overview and comparison of different algorithms for the tessellation of NURBS surfaces. The process of tessellating a NURBS surface entails transforming its continuous surface into a finite number of triangles. Due to the unwieldy nature of NURBS surfaces, they must often first be transformed into an intermediate representation to be used. As such, the topic of NURBS surface tessellation has been widely studied within the context of computer graphics and computer-assisted design. However, little is known about the best practices for tessellating NURBS surfaces in tactile physics simulations.

For the purposes of this research, tessellation algorithms that generate a mesh within a user-specified tolerance are of particular interest. This is because any imperfections in the mesh will eventually not be perceivable by humans, given a small enough tolerance. Finding the exact tolerance at which this happens is crucial for minimizing mesh complexity without affecting smoothness.

The most common strategies for NURBS tessellation can broadly be divided into two categories [11]:

- **Global Spacing Tessellation**: Mesh vertices are generated from a uniform grid in parametric space. The maximum curvature of the NURBS surface decides geometry density throughout the entire mesh [12; 13].
- **Adaptive Tessellation**: Geometry is adaptively assigned to areas with greater amounts of curvature. Common strategies for adaptive tessellation consist of repeated subdivision of the surface [14; 15]. Alternative approaches using bubble dynamics for vertex generation also exist [16].

Each of these categories of tessellation algorithms has its advantages and disadvantages. Adaptive tessellation avoids assigning excessive geometry to flat areas and therefore creates more efficient meshes. However, global spacing tessellation algorithms are generally easier to implement [8]. Additionally, global spacing algorithms generally avoid creating long and skinny triangles, which negatively affect the numerical stability of many computational routines performed on triangles [13].

3 Methodology

This section will discuss the manner in which the research for this paper was conducted. It will justify the choices made and highlight the issues encountered throughout the entire process.

3.1 Choice of NURBS Surfaces for Smooth Object Representation

NURBS surfaces are not immediately the most natural choice for object representations from the context of remote environment discovery. Computer vision algorithms typically discover the world by representing it through a point cloud. However, for the purposes of this research, we are working under two critical assumptions:

- (1) The problem of discovering the remote environment has been solved perfectly.
- (2) The shape of discovered remote objects is provided to the local model through NURBS surfaces.

Initially, an attempt was made to directly use NURBS surfaces inside of a physics engine. However, after some consideration, full integration of NURBS surfaces in a tactile physics engine was deemed infeasible. Therefore, the choice was made to focus on accurately turning NURBS surfaces

into a mesh, through a process called tessellation. This has a number of advantages. To start off, meshes profit from native support in most physics engines. Additionally, haptic rendering on meshes is well documented [17]. Finally, a mesh will eventually become indistinguishable from the NURBS surface it approximates, given enough geometry. Finally, the haptic rendering of meshes is a well-documented subject matter.

For the evaluation of the generated meshes, the choice was made to analyze the mesh complexity and perceived smoothness. These two properties were considered to be especially important for the purposes of the tactile internet for several reasons. Firstly, complex meshes negatively affect the frequency at which the local models can operate. To ensure that complex remote environments can be simulated, meshes should not be more complex than reasonably needed. Secondly, meshes should have enough geometry to resemble the original surface and be perceived as smooth. Therefore, a trade-off between mesh complexity and smoothness lies central to the problem of finding a suitable way to tessellate NURBS surfaces for use in tactile internet.

3.2 Tessellation Method

A global spacing algorithm was chosen as the tessellation algorithm to be used for the purposes of this paper. This algorithm was chosen due to its simplicity of implementation and it having reasonable performance [8]. An attempt was also made to implement an adaptive tessellation algorithm such as described by Piegl and Tiller [14]. This algorithm would in theory be able to create lower-complexity meshes for the same tolerance. However, due to the complexity of the algorithm, it could not be implemented within the time allotted for the research project.

The global spacing tessellation algorithm is based on the idea of finding a maximum edge length λ for a user-specified tolerance ϵ to triangulate the parametric space. We want λ, such that any line segment AB of length λ in parameter space will have a corresponding straight line from $S(A)$ to $S(B)$ that deviates from the NURBS surface by at most tolerance ϵ, where, S is the surface function of the NURBS surface. Using λ, the parametric space is uniformly divided up into square regions. The corners of these regions are then used to calculate the vertices of the mesh.

Obtaining λ

For determining a maximum edge size λ, the method described by Piegl and Richard was used [13]. The maximum edge length is determined by the formula

$$\lambda = 3\sqrt{\frac{\epsilon}{D_1 + 2D_2 + D_3}}, \tag{1}$$

where

$$D_1 = \sup_{u,v} ||S_{uu}(u,v)||, \tag{2}$$

$$D_2 = \sup_{u,v} ||S_{uv}(u,v)||, \tag{3}$$

$$D_3 = \sup_{u,v} ||S_{vv}(u,v)||. \tag{4}$$

In other words, an upper bound on the second-order partial derivatives of the NURBS surface must be obtained to calculate a maximum edge length λ. Unfortunately, no analytical solution for finding the exact bounds is currently known [18].

Initially, the method described by Piegl and Richard was used for calculating the surface derivative bounds. Since the derivative of a NURBS surface is again a NURBS surface, we can calculate the control points associated with the derivative surfaces. As the control net of a NURBS surface forms a convex hull around the surface, it can be used to bound the surface itself [13].

However, this method was found to have a number of restrictive issues. To start, it was found to not be compatible with all NURBS surfaces. Surfaces having repeating knots with a multiplicity of the degree minus one would cause a division by zero in the calculation of derivative control points. Knots of this multiplicity are valid for NURBS surfaces and convenient for creating surfaces such as spheres. Additionally, the bounds on D_1, D_2, D_3 were found to not be tight for surfaces with a low amount of control points.

To solve the first issue, a technique called knot insertion was used. In essence, this allows the surface to be split up into multiple sub-surfaces, which each contain no repeated knots. We can then determine the bounds on the partial derivatives of each sub-surface separately, and use the maximum of these values for the calculation of λ.

To improve the tightness of the calculated bounds on D_1, D_2, D_3, a technique called knot refinement was used. Through knot refinement, it is possible to add extra control points to a NURBS surface, while retaining its shape. While added control points make it more expensive to estimate the partial derivative bound, they also make the control net better approximate the NURBS surface. In turn, this allows for tighter bounds on the partial derivatives of the surface.

Mesh Generation

Once the maximum edge length λ in parameter space has been found, it can be used to generate vertices for our mesh. To do this, a uniform grid of points in parameter space is used to create vertices. We calculate the number of grid points in the u and v directions of the parametric space by

$$n_u = \lceil \frac{\max u_i}{\lambda} \rceil + 1 \tag{5}$$

$$n_v = \lceil \frac{\max v_i}{\lambda} \rceil + 1. \tag{6}$$

To generate the final vertex corresponding to a point (u_0, v_0) in parameter space, we can use the surface function $S(u,v)$ to map the point onto the NURBS surface.

After generating a grid of vertices, the faces of the mesh are constructed. Each rectangle consisting of vertices P_0, P_1, P_2, P_3 in the grid, is used to generate two triangular faces: $\Delta P_0 P_1 P_2$ and $\Delta P_0 P_2 P_3$. Crucially, the hypotenuse of these triangles has exactly length λ, due to their remaining sides having a length of $\frac{1}{\sqrt{2}}\lambda$.

The final output of the tessellation algorithm consists of a mesh stored in the Wavefront file format. This choice was made due to the simplicity, versatility, and universal support of the format [19]. Crucially, the mesh file would allow the

approximated NURBS surface to be simulated in a tactile physics simulation. In addition, the exported mesh file allowed for detailed analysis and rendering of the mesh, which would be needed to produce our results.

3.3 Investigating Perceived Smoothness

Perhaps the most important characteristic of meshes generated through tessellation of NURBS surfaces, is how smooth they feel to humans. Ideally, we want surfaces to be tessellated at such a tolerance ϵ that surfaces are just barely perceived as smooth. This minimizes mesh complexity, which is desirable for ensuring a $1000\,\mathrm{Hz}$ refresh rate for tactile physics engines [20].

Finding an objective metric for this smoothness proved to not be possible, as humans set the yardstick for what mesh quality is considered smooth. Instead of trying to find a way around this subjectiveness, it was embraced. Meshes generated through the implemented tessellation algorithm would need to be put to the test in a survey. In this survey, participants would be asked to categorize meshes based on their perceived smoothness. By varying the tolerance of meshes, an optimal tolerance for both high smoothness and minimal mesh complexity could be obtained. The exact details of the survey and the manner in which the survey was conducted will be explained in the *experimental setup* section of this paper.

To find a good range of tolerances to survey, previous research in the area of smoothness perception was investigated. Here, it is suggested that bumps of $0.1\,\mathrm{mm}$ are the threshold for surfaces being perceived as smooth [21; 22]. As a precaution, the lowest tolerance used in the survey was set to half of this value, at $0.05\,\mathrm{mm}$.

4 Experimental Setup

The following sections describe what experiments were performed to obtain results. It starts off by discussing the types of NURBS surfaces used for the purpose of tessellation. Then, the survey used to conduct our user study on smoothness perception is explained in detail.

4.1 Mesh Complexity

To get an idea of the effectiveness of the implemented tessellation algorithm, a number of NURBS surfaces were needed for testing purposes. While NURBS surfaces could be obtained by creating a model in 3D modeling software, doing so is generally tricky and time-consuming. The amount of NURBS surfaces found online was also limited. As such, an alternative approach to obtaining surfaces was chosen.

The surfaces chosen for testing the tessellation algorithm were surfaces of revolution. Such surfaces were obtained by rotating a two-dimensional NURBS curve around an axis [23]. As such, a wide variety of shapes could be obtained without the need for complex modeling.

For the purposes of this paper, four surfaces in the shape of a sphere, torus, bullet, and hourglass were used. These surfaces were chosen for the variety of their respective curvatures. For instance, a sphere has uniform curvature throughout its surface. Meanwhile, the hourglass surface features both areas with a great or tiny amount of curvature. As curvature is hugely important for the amount of geometry needed for tessellation, the choice of shapes would give a good overview of the performance of the tessellation algorithm in different scenarios.

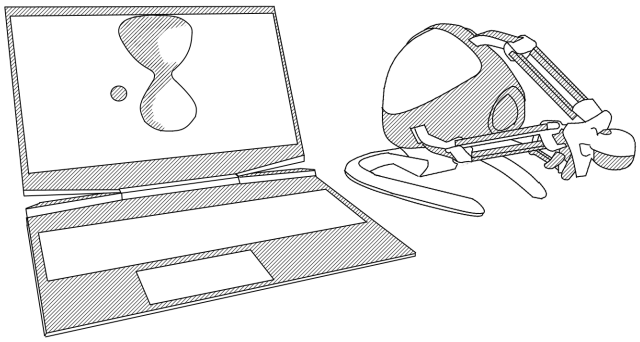


Figure 3: Illustration of the hardware setup used to conduct the survey. On the right, the Novint Falcon controller used to interact with meshes is shown. On the left, a laptop screen displaying the survey application can be seen.

4.2 Perceived Smoothness Survey

To investigate how high the tolerance ϵ could be while still preserving mesh smoothness, a survey application was developed. This application allowed participants to experience the haptics generated from meshes, and subsequently grade their perceived smoothness. For haptic generation, the CHAI3D [1] framework was used. A Novint Falcon[2] haptic controller was used to allow participants to interact with the mesh and feel the resulting haptic feedback. An illustration of the survey setup can be seen in Fig. 3.

The survey was conducted as follows. Participants were shown meshes generated from multiple surfaces at varying tolerances. Four surfaces were tessellated to acquire these meshes: an hourglass, a sphere, a torus, and a bullet surface. These surfaces were then tessellated with global spacing tessellation at tolerances of $\epsilon = 0.5, 0.2, 0.1, 0.05, 0.02, 0.01, 0.005$ and 0.002. The meshes of each shape were presented to the participant in decreasing order of tolerance. Participants were then asked to feel the mesh and grade its smoothness. Participants were always shown a high-resolution mesh of these surfaces on a display so that visuals could not impact the perceived smoothness. Fig. 4 shows a screenshot from the final survey application.

Participants were then asked to classify the presented meshes into one of three categories. These categories were defined as follows:

- **Angular**: The participant can feel clear edges within the mesh. Smooth movement across the surface is not possible, and the haptic cursor can get stuck behind edges of the geometry.
- **Coarse**: The participant cannot feel clear edges within the mesh anymore. The surface can be felt as if having

[1] chai3d.org

[2] novint.com

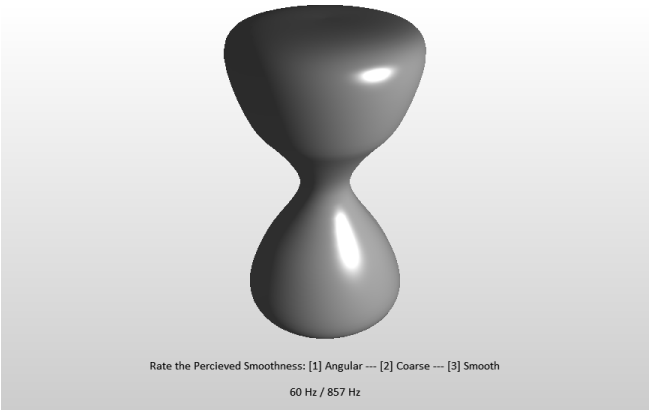


Figure 4: Screenshot from within the application used to survey the perceived smoothness of meshes. Participants are shown a high-resolution render of a shape, while a lower-resolution mesh is used to generate the haptic feedback to be evaluated by participants.

Tolerance ϵ	Physical Distance
0.5	$12.5\,\mathrm{mm}$
0.2	$5\,\mathrm{mm}$
0.1	$2.5\,\mathrm{mm}$
0.05	$1.25\,\mathrm{mm}$
0.02	$0.5\,\mathrm{mm}$
0.01	$0.25\,\mathrm{mm}$
0.005	$0.125\,\mathrm{mm}$
0.002	$0.05\,\mathrm{mm}$

Table 1: Conversion of digital tolerance ϵ to real-world tolerance ϵ_R for all tolerances used during the survey.

a coarse texture, with many small bumps. These bumps do not impede movement over the surface.

- **Smooth**: The participant can smoothly move the haptic cursor over the surface, without perceiving small bumps on the surface of the mesh.

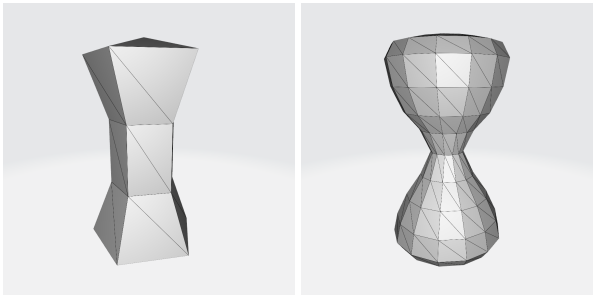
The use of a haptic controller gives the used tolerances a real-world context. The Novint Falcon controller can move within a sphere with a diameter of $10\,\mathrm{cm}$. This range of movement is mapped to a sphere with a diameter of 1 unit inside of the survey application. Subsequently, models were imported into the application at a scale of 0.25. This allows us to calculate the real-world tolerance ϵ_R in millimeters from the digital tolerance ϵ with the formula

$$\epsilon_R = \frac{100\,\mathrm{mm}}{4}\epsilon.$$

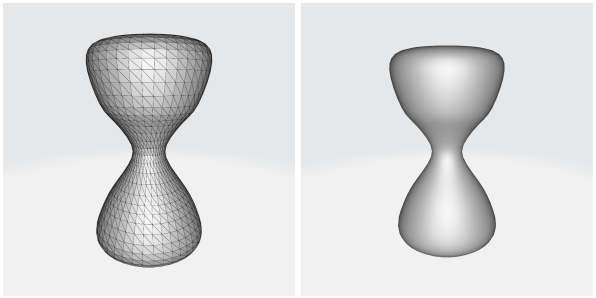
Table 1 shows the conversion of the tolerances ϵ used in the survey to the real-world tolerance ϵ_R.

5 Results

The following section will give an analysis of the implemented tessellation algorithm. First, we investigate the complexity of the resulting meshes. Secondly, we discuss the perceived smoothness of these meshes at different mesh tolerances, which was investigated through a user study.



(a) Mesh at tolerance $\epsilon = 0.5$ (b) Mesh at tolerance $\epsilon = 0.01$



(c) Mesh at tolerance $\epsilon = 0.005$ (d) Mesh at tolerance $\epsilon = 0$

Figure 5: Meshes obtained from the tessellation of an hourglass NURBS surface with the global spacing tessellation algorithm at varying tolerances ϵ, along with the true shape of the NURBS surface that these meshes approximate. As can be seen, the resulting mesh starts to more closely resemble the true surface as the tolerance decreases.

5.1 Mesh Complexity

Fig. 5 shows the mesh resulting from tessellating the hourglass surface at different user-specified tolerances ϵ. Visually speaking, the tessellation algorithm behaves as expected. We can observe that for a high tolerance ϵ, the resulting mesh has a low complexity. This makes sense, as the higher the allowed error in a mesh is, the less geometry is needed to represent a surface. On the other hand, the mesh starts to resemble the NURBS surface increasingly more accurately as the tolerance decreases. Of course, this comes at the cost of a higher required polygon count for the mesh. Eventually, as the tolerance ϵ approaches 0, we would theoretically approach a perfect mesh of the NURBS surface, with an infinite polygon count.

The effect of the tolerance ϵ on the accuracy of the generated mesh can be seen even more clearly in fig. 6, where a 2-dimensional cross-section of the hourglass mesh is examined. As can be seen, the mesh approximation is allowed to deviate by a significant margin for a high tolerance ϵ. This is especially apparent near the top and bottom of the hourglass. Again, lower tolerance can be seen to increase how well the mesh approximation matches the real surface.

The complexity cost of lowering the tolerance with regards to the vertex count can be seen in Fig. 7. Here, mesh complexity is expressed in term of the polygon count. The spherical NURBS surface represents the best case for the tessellation algorithm, with the lowest mesh complexities for any

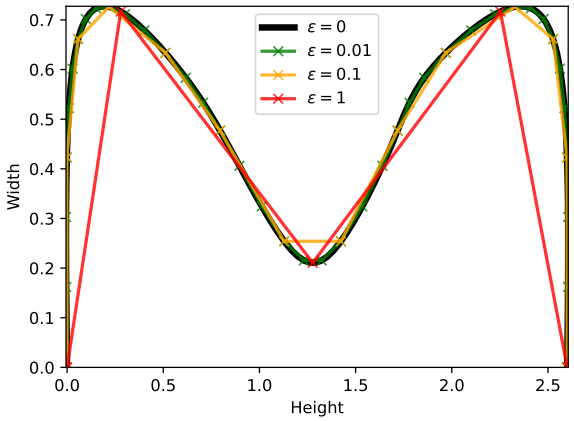


Figure 6: Cross section of the meshes obtained from the tessellation of an hourglass NURBS surface. The shape of the real surface is shown in black, while the mesh approximations at different tolerances ϵ are shown as colored dashed lines drawn on top.

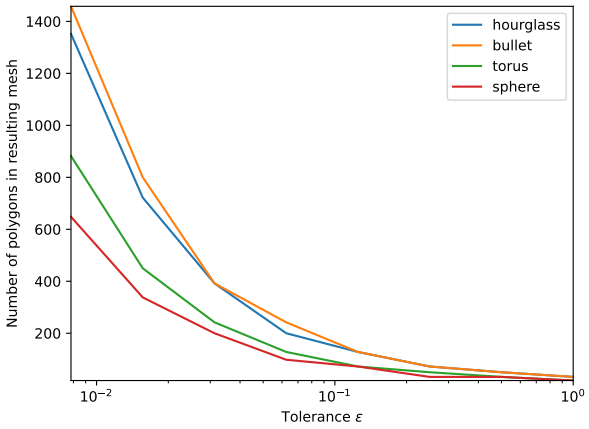


Figure 7: Plot of the mesh complexity as a function of the supplied tolerance ϵ for the global spacing tessellation algorithm. Here, mesh complexity is expressed as the number of polygons in the mesh. To generate the meshes, NURBS surfaces in the shape of a sphere, torus, bullet, and hourglass were used. More irregular surfaces, such as the bullet and hourglass, can be seen to lead to more complex meshes.

given tolerance. In general, the meshes needed to approximate the hourglass and bullet surfaces are more complex by around a factor of 2 than those of the sphere. Finally, we can see that the torus surface has a complexity somewhere in the middle. From the figure, a roughly inversely proportional relation between the tolerance ϵ and the resulting mesh complexity seems to exist, with the mesh complexity doubling when the tolerance is halved.

The findings exemplify the need for careful selection of the tolerance when generating meshes for use in tactile physics engines. As Fig. 7 shows, a small decrease in tolerance can drastically increase the number of polygons present in the resulting mesh. This is undesirable, as complex meshes slow down the tactile physics simulation, which must run at 1000 Hz. As such, great care must be taken for finding a precise tolerance ϵ, at which the imperfections of the resulting mesh are just barely imperceivable.

Additionally, the mesh complexities in Fig. 7 show a drawback of the chosen tessellation method. As global spacing only considers the upper bounds on the partial derivatives of the NURBS surface, the worst-case curvature determines the edge length λ for the entire mesh. As a result, excessive amounts of geometry are used in the flatter parts of a surface. This is reflected in the mesh complexities of the hourglass and bullet surfaces, which have a very non-uniform curvature. Both surfaces have significantly more complex meshes than those of the sphere and torus surfaces, which have a more uniform curvature.

Fig. 8 demonstrates that the error of meshes generated by the tessellation algorithm does not exceed the tolerance ϵ. Here mesh error is defined as the Hausdorff distance between the NURBS surface and its corresponding mesh. The Hausdorf distance is estimated trough $n = 10,000$ points taken uniformly on the NURBS surface. For most surfaces, the maximum error is almost an order of magnitude lower than the tolerance. The findings confirm that the tessellated meshes abide by the supplied tolerance.

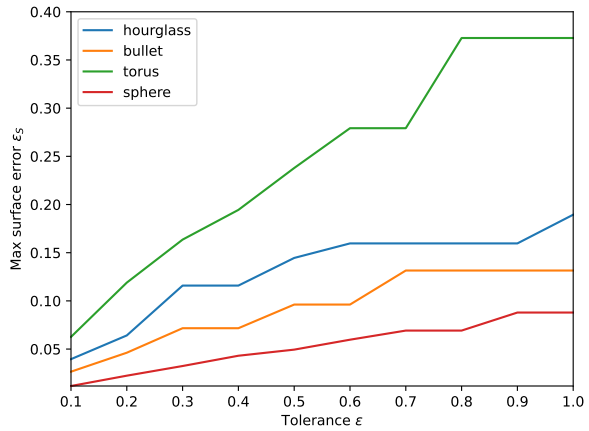
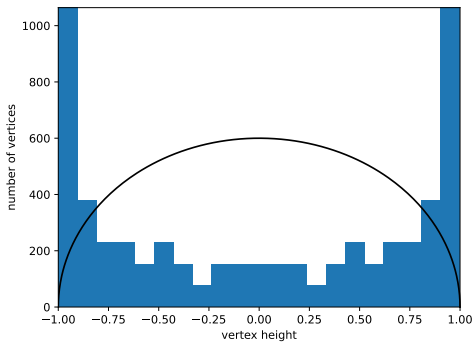
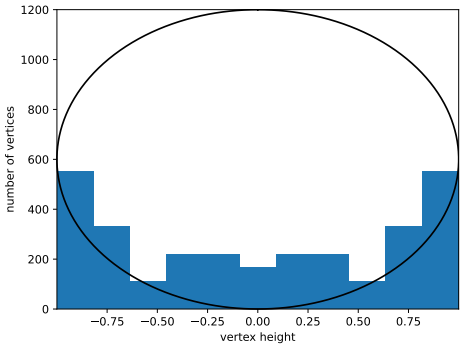


Figure 8: The maximum error found within meshes of different NURBS surfaces at varying tolerances. Here the maximum error was determined as the Hausdorff distance between the mesh and the original NURBS surface. As can be seen, the mesh error does not exceed the tolerance ϵ at any point.

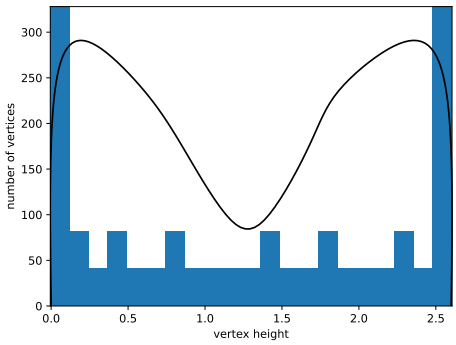
To see how much geometry is used to represent the different parts of the investigated NURBS surfaces, the vertices of the mesh were grouped based on their height within the mesh. The resulting number of vertices present at different heights was collected as a histogram for each of the investigated shapes. As a visual guide to the location of geometry within the surface, a cross-section of the original NURBS surfaces was overlayed on the histogram. The resulting figure is displayed in Fig. 9. The figure shows geometry being most concentrated near the top and bottom of the generated



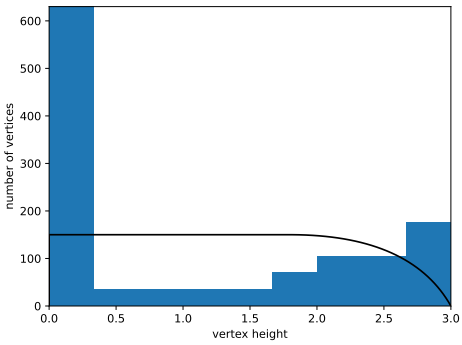
(a) Geometry distribution of sphere



(b) Geometry distribution of torus



(c) Geometry distribution of hourglass



(d) Geometry distribution of bullet

Figure 9: Histogram of the concentration of geometry on meshes generated at a tolerance of $\epsilon = 0.01$ for different NURBS surfaces. Here, the amount of geometry is expressed by the number of vertices. The outline of the surface cross-section is overlayed in the figures, to indicate the spatial location of the vertices on the surface. Geometry can be seen to be most concentrated near the top and bottom of the mesh.

meshes. This also holds for NURBS torus and hourglass, whose bottom areas are assigned a lot of geometry despite being nearly flat. However, the midsection of the bullet is represented with very little geometry. For all other shapes, the amount of geometry used for the middle section is fairly uniform throughout.

The distribution of vertices highlights another drawback of the chosen algorithm. Namely, that geometry is not necessarily distributed evenly over the surface. As can be seen in Fig. 9, the amount of vertices assigned to the poles of the sphere surfaces is large compared to the number of vertices assigned to its midsection. However, the midsection of a sphere makes up the majority of the surface area of a sphere, while the poles only represent a fraction of the total area. Therefore, a mismatch between surface area and geometry concentration exists for the global tessellation algorithm.

All in all, the global spacing algorithm seems to function reasonably well but does have its fair share of issues. The algorithm functions as it's supposed to and generated meshes strictly stay within a distance ϵ of the actual surface. However, surfaces with irregular curvature in particular are assigned too much geometry in low curvature areas. Additionally, excessive geometry is used near the poles of the investigated surfaces of revolution. Due to the necessity of keeping the complexity of meshes used in tactile physics simulations to an absolute minimum, this is undesirable.

5.2 Perceived Smoothness

The perceived smoothness survey outlined in section 4.2 of this paper was conducted on $n = 21$ participants. However, it should be noted that primarily young men experienced with computers were surveyed. The group of participants is therefore not representative of the general population.

The results from this survey have been compiled in Fig. 10. As can be seen, meshes with a tolerance below $\epsilon = 1.25\,\mathrm{mm}$ were rarely perceived as being smooth or coarse. Most surfaces only start to be perceived as smooth by a majority of participants at tolerances as low as $\epsilon = 0.05\,\mathrm{mm}$. The bullet surface seems to be the exception to this, already being perceived as smooth at a tolerance of $\epsilon = 0.125\,\mathrm{mm}$ by most participants. This is likely due to the relatively low amount of curvature within the bullet's surface.

Despite meshes needing a very low tolerance to be perceived as smooth, a significantly higher tolerance is already sufficient for the perception of a coarse surface. As can be seen in Fig. 10, a tolerance of just $\epsilon = 0.5\,\mathrm{mm}$ is enough for the surface to be perceived as coarse for a majority of participants. Furthermore, a tolerance of $\epsilon = 0.25\,\mathrm{mm}$ is suffi-

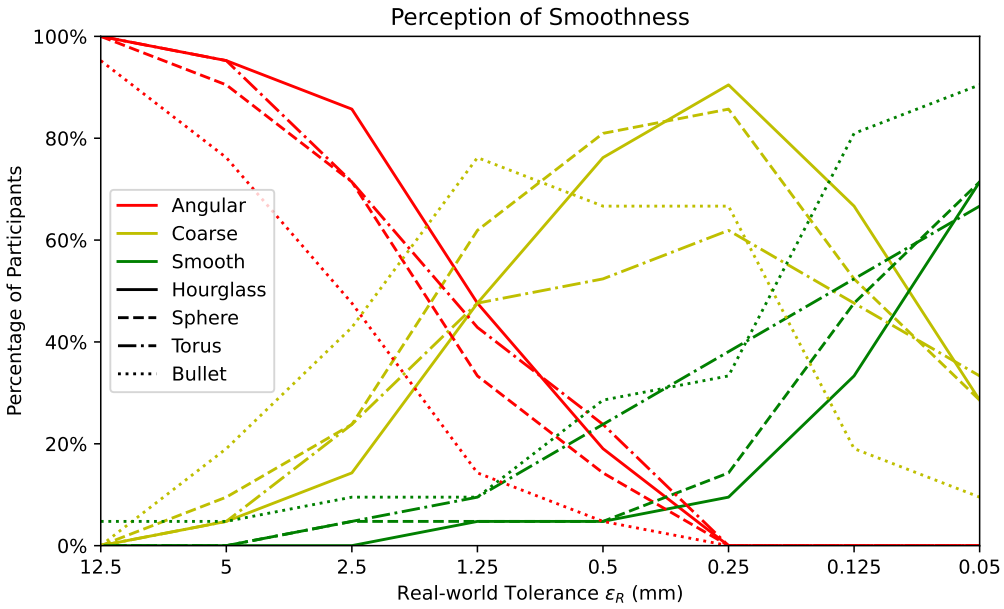


Figure 10: A diagram showing the results of the perceived smoothness survey, conducted on $n = 21$ participants. The different line colors indicate how smooth meshes were perceived as, while the line style indicates the shape of the mesh. Lowering the tolerance can be seen to improve the perceived smoothness of meshes. At a tolerance of $\epsilon = 0.05\,\mathrm{mm}$, a clear majority of participants perceived the mesh as smooth for all shapes.

cient for participants to exclusively rate surfaces as coarse or smooth. Once again, the bullet surface can be to be perceived as coarse at lower tolerances than other surfaces.

Interestingly, some participants of the survey perceived meshes to not be smooth even at the lowest tolerance. This is especially apparent for the sphere, torus, and hourglass surfaces. These findings suggest that the tolerance required for smooth object representation lies even lower than the investigated values. There is also a possibility the hardware is reaching its limits of spatial resolution at this point. In any case, a more comprehensive survey would be needed to find a tolerance that is satisfactory for all participants.

All in all, the results of the survey suggest that for proper representation of smooth surfaces a tolerance of $0.05\,\mathrm{mm}$ or lower is required. However, due to the significant increase in mesh complexity caused by lowering the tolerance, it should not be set lower than necessary. The optimal tolerance is likely highly dependent on the context of the tactile internet application. In general, a tolerance of $\epsilon = 0.05\,\mathrm{mm}$ should be enough for most purposes of tactile internet, as a clear majority of participants start perceiving surfaces as smooth here. However, precision tactile internet applications, such as remote surgery, would likely benefit from a lower tolerance. For applications where only shape, but not the smooth texture of an object has to be represented, a tolerance of $\epsilon = 0.25\,\mathrm{mm}$ should be used to reduce the burden of the tactile physics engine.

6 Responsible Research

6.1 Use of Participants

For the smoothness perception survey, a number of steps were taken to ensure the ethical use of participants. Firstly, great care was taken to ensure informed consent. Participants were first clearly explained what the purpose of the survey was. Participants were also made aware of the fashion in which their perception of smoothness would be used. Secondly, the privacy and anonymity of participants were ensured. Absolutely no personal data was used in the survey, and it would be impossible to trace obtained data back to a specific participant.

Additionally, an effort was made to respect the spare time of participants. Most participants consisted of fellow bachelor students working hard on their respective research, with little time to spare. Taking up a large amount of their time could impede their research or studies. Therefore, the survey was designed to be as streamlined as possible so it could be completed within 5 to 10 minutes. In practice, most participants managed to complete the survey within this time.

6.2 Reproducibility

Great care was taken during the research discussed in this paper to ensure its results are reproducible. All code used for this paper, including tessellation algorithms, graphing scripts, and surface generators, have been made publicly available through a repository. This repository includes detailed instructions on how to run the code. Anyone with moderate programming knowledge should be able to follow along with the provided instructions with ease.

7 Discussion

In this section, we will place the obtained results in a broader context. To do this, results will be compared to similar works in the field. This will give us an intuition of how accurate our findings are.

Directly comparing obtained results to other papers on NURBS tessellation was difficult, as the NURBS surfaces used in such papers are often not explicitly given. However, certain aspects of the behavior of the algorithm could be verified. Firstly, the effect of lowering the tolerance on mesh complexity. In the paper our tessellation algorithm is based on, Piegl and Richard find the relation between the tolerance and the number of polygons to be inversely proportional, which matches our findings. Furthermore, our research found that the maximum error for the generated meshes was approximately one-tenth of the tolerance in most cases. This is also in accordance with the error found in the original paper [13].

Within the field of haptic rendering, the minimum height of imperfections in a mesh that can still be perceived by humans is commonly considered to be around $100\,\mu\mathrm{m}$. However, it should be considered that the user studies conducted to find these values typically only ask participants to rate a mesh as either smooth or non-smooth [21; 22]. Meanwhile, our own smoothness perception survey asked participants to classify a mesh into one of three categories. A similar study conducted in the field of biology suggests grooves with a width of up to $40\,\mu\mathrm{m}$ are still perceivable by humans [24].

All considered, the found tolerance value for perceived smoothness agrees well with current knowledge. The discovered tolerance of $\epsilon = 50\,\mu\mathrm{m}$ is well within the range of $100\,\mu\mathrm{m}$ to $40\,\mu\mathrm{m}$ suggested by literature.

8 Future Work

This section will highlight a number of avenues that future research might take to expand on the findings of this paper. Given a longer time frame to continue research, these would represent the next steps taken to improve the findings of this paper.

- *Alternative algorithms* for the tessellation of NURBS surfaces have not exhaustively been tested within this paper. Other approaches, such as adaptive subdivision [14] and bubble mesh generation [16], might prove a better fit for the given situation. As such, a more comprehensive comparison of algorithms could yield an improved process for NURBS surface tessellation for use in tactile physics simulations.
- *Force shading* might also prove a promising avenue for smooth object representation in tactile physics simulations. The notion of force shading is analogous to the way shading is commonly used to improve the appearance of three-dimensional objects in the field of computer graphics. Force shading has been shown to improve perceived surface smoothness without requiring extra geometry [25]. Combining force shading with the techniques described in this paper may allow for lower mesh complexities with no degradation in perceived smoothness.
- *Use of more realistic surfaces* for tessellation would allow for a more complete understanding of NURBS tessellation in practice. Only relatively simple surfaces were investigated in this paper. However, the real-world objects we want to represent in the local physics model might be much more complex. Analyzing the tessellation of such objects would allow us to understand the performance of tessellation algorithms in a realistic context.

9 Conclusion

Tactile internet is an emerging internet paradigm, allowing for precise control of robotic devices at a distance. To make the haptics used in tactile internet feel convincing over long distances, proper representation of smoothly curved objects is necessary. NURBS were found to be a powerful model for representing such objects but were deemed incompatible with current tactile physics engines. As such, this paper investigated the effectiveness of converting NURBS surfaces to a mesh through a global spacing tessellation algorithm. An analysis of mesh complexity was performed, and a user study was conducted to find mesh tolerances sufficient for perceived smoothness.

A global spacing tessellation algorithm was implemented and tested on a number of NURBS surfaces. The algorithm generally behaved as expected, needing greatly more geometry to approximate a surface as the mesh tolerance decreases. While the meshes generated with this algorithm stayed within the user-specified tolerance, geometry was excessively concentrated near the top and bottom of the tested surfaces. The algorithm also seems to struggle more with surfaces where curvature suddenly changes.

Our user study on perceived smoothness found that meshes tessellated at a tolerance of $\epsilon = 0.05\,\mathrm{mm}$ were sufficient to be perceived as smooth by a majority of participants. Lowering the tolerance further to increase perceived smoothness is not recommended unless a Tactile Internet application requires extreme precision. Otherwise, the dramatic increase in mesh complexity as tolerance decreases will unnecessarily burden the tactile physics simulation.

On the whole, tessellation of NURBS surfaces through the global tessellation algorithm at a tolerance of $\epsilon = 0.05\,\mathrm{mm}$ seems to be adequate for most purposes of tactile internet.

References

[1] Gerhard Fettweis. The Tactile Internet: Applications and Challenges. *IEEE Vehicular Technology Magazine*, 9(1):64–70, 3 2014.

[2] Vineet Gokhale, Mohamad Eid, Kees Kroep, Ramjee Prasad, and Vijay M. Rao. Toward Enabling High-Five Over WiFi: A Tactile Internet Paradigm. *IEEE Communications Magazine*, 59(12):90–96, 12 2021.

[3] Kurian Polachan, Joydeep Pal, Chandramani Singh, T. V. Prabhakar, and Fernando A. Kuipers. TCPSbed: A Modular Testbed for Tactile Internet-Based Cyber-Physical Systems. *IEEE ACM Transactions on Networking*, pages 1–16, 1 2021.

[4] Joseph J. LaViola. A discussion of cybersickness in virtual environments. *ACM Sigchi Bulletin*, 32(1):47–56, 1 2000.

[5] Frank H. P. Fitzek, Shu-Chen Li, Stefanie Speidel, Thorsten Strufe, and Patrick Seeling. Frontiers of Transdisciplinary Research in Tactile Internet with Human-in-the-Loop. 9 2021.

[6] Shusen Zheng, Burak Cizmeci, Clemens Schuwerk, and Eckehard Steinbach. Model-Mediated Teleoperation: Toward Stable and Transparent Teleoperation Systems. *IEEE Access*, 4:425–449, 1 2016.

[7] Les A. Piegl. On NURBS: a survey. *IEEE Computer Graphics and Applications*, 11(1):55–71, 1 1991.

[8] Shawn P. Austin, Robert B. Jerard, and Robert G. Drysdale. Comparison of discretization algorithms for nurbs surfaces with application to numerically controlled machining. *Computer Aided Design*, 29(1):71–83, 1 1997.

[9] Elaine Cohen, Tom Lyche, and Richard F. Riesenfeld. Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics. *Computer Graphics and Image Processing*, 14(2):87–111, 10 1980.

[10] Thomas J. R. Hughes, J. A. Cottrell, and Yuri Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39-41):4135–4195, 10 2005.

[11] Subodh Kumar and Dinesh Manocha. Efficient rendering of trimmed nurbs surfaces. *Computer Aided Design*, 27(7):509–521, 7 1995.

[12] Bernd Hamann and Po-Yu Tsai. A tessellation algorithm for the representation of trimmed nurbs surfaces with arbitrary trimming curves. *Computer Aided Design*, 28(6-7):461–472, 6 1996.

[13] Leslie A Piegl and Arnaud M. Richard. Tessellating trimmed nurbs surfaces. *Computer-Aided Design*, 27(1):16–26, 1 1995.

[14] Les A. Piegl and Wayne Tiller. Geometry-based triangulation of trimmed NURBS surfaces. *Computer Aided Design*, 30(1):11–18, 1 1998.

[15] G. V. V. Ravi Kumar, Prabha Srinivasan, K.G. Shastry, and B. G. Prakash. Geometry based triangulation of multiple trimmed NURBS surfaces. *Computer Aided Design*, 33(6):439–454, 5 2001.

[16] Qisheng Wang, Boqing Gao, and Hui Wu. Triangular mesh generation on free-form surfaces based on bubble dynamics simulation. *Engineering Computations*, 36(2):646–663, 1 2019.

[17] Stephen D. Laycock and Andy M. Day. A Survey of Haptic Rendering Techniques. *Computer Graphics Forum*, 26(1):50–65, 3 2007.

[18] Ye Tian, Tao Ning, Jixing Li, Jianmin Zheng, and Zhitong Chen. An Improvement on the Upper Bounds of the Partial Derivatives of NURBS Surfaces. *Mathematics*, 8(8):1382, 8 2020.

[19] Keith Rule. *3D Graphics File Formats*. Addison-Wesley, 1 1996.

[20] Pingjun Xia. New advances for haptic rendering: state of the art. *The Visual Computer*, 34(2):271–287, 10 2016.

[21] Naoya Asamura, Takenao Shinohara, Y Tojo, Nobuyoshi Koshida, and Hiroyuki Shinoda. Necessary spatial resolution for realistic tactile feeling display. 5 2001.

[22] Jian Zhang, Shahram Payandeh, and John Dill. Levels of detail in reducing cost of haptic rendering: a preliminary user study. 12 2003.

[23] Les A. Piegl and Wayne Tiller. *The NURBS Book*. 1 1997.

[24] John E. Morley, Antony W. Goodwin, and Ian Darian-Smith. Tactile discrimination of gratings. *Experimental Brain Research*, 49(2), 2 1983.

[25] Hugh Brian. Morgenbesser and Mandayam A. Srinivasan. Force shading for haptic shape perception. *Journal of Dynamics Systems, Measurement, and Control*, 58:407–412, 12 1996.