2007

Bezíer Surface Generation of the Patella

Dale A. Patrick
Wright State University

Follow this and additional works at: http://corescholar.libraries.wright.edu/etd_all
Part of the Electrical and Computer Engineering Commons

Repository Citation
BEZIER SURFACE GENERATION OF THE PATELLA

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering

By

Dale A. Patrick
BSEE University of Toledo, 2002

2007
Wright State University
I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Dale Allen Patrick ENTITLED Bezier Surface Generation of the Patella BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in Engineering.

Kefu Xue, Ph.D.
Thesis Director

Fred Garber, Ph. D.
Department Chair

Committee on
Final Examination

Kefu, Xue, Ph.D.

Lang Hong, Ph. D.

Arnab K. Shaw, Ph.D.

Joseph F. Thomas, Jr., Ph. D.
Dean, School of Graduate Studies
ABSTRACT


A method to generate control points for a given collection of data points is developed. The shape characteristics of the patella are used as a case study. The developed computer graphic techniques are demonstrated using the given data of the 3-d patella model. Generation of curved surfaces in drafting has evolved over the years from draftsman who used tools such as French curves to today’s drafting being well served by being able to apply numerical analysis on the ever increasing capabilities of computer. As an everyday observed example, nature very rarely produces two shapes that are equal to another. It is also very infrequent that straight lines or flat planes are produced as a natural occurrence. Curved three dimensional surfaces such as the patella are particularly difficult to model because as with most things found in nature each patella has unique and complex features. Applying a pseudo-inverse matrix technique to generate Bezier surface patch control points for an existing set of collected three dimensional data points is introduced to describe such features. Affine transforms will be applied to the control points to adjust the scale of the original shape. Lastly, the error will be measured in order to describe the effect the Bezier surface patch generation and affine transforms have compared next to the given collection of data points.
# TABLE OF CONTENTS

I. INTRODUCTION .......................................................................................................................... 1  
  1.1 BACKGROUND .............................................................................................................................. 1  
  1.2 APPROACH ..................................................................................................................................... 3  

II. LITERATURE REVIEW ............................................................................................................. 6  
  2.1 2-D BEZIER CURVES ................................................................................................................... 6  
  2.2 3-D BEZIER SURFACES .............................................................................................................. 14  
  2.3 THE GEOMETRIC CENTER AND THE CENTER OF MASS ....................................................... 27  
  2.4 PSEUDO INVERSE MATRIX ......................................................................................................... 27  
  2.5 MEASURING ERROR .................................................................................................................. 28  
  2.6 AFFINE TRANSFORMS .............................................................................................................. 29  

III. DEVELOPMENT .................................................................................................................. 32  
  3.1 INITIAL RAW DATA .................................................................................................................... 32  
  3.2 THE PSEUDO INVERSE TECHNIQUE ...................................................................................... 34  
  3.3 SINGLE CONTROL POINT MESH OF THE PATELLA .......................................................... 35  
  3.4 SINGLE CONTROL POINT MESH LAYER OF THE PATELLA ............................................. 38  
  3.5 BEZIER SURFACE GENERATING ALGORITHM ....................................................................... 40  
  3.6 ENTIRE BEZIER SURFACE OF THE PATELLA ....................................................................... 44  
  3.7 AFFINE TRANSFORMATION ...................................................................................................... 46  
  3.8 ERROR MEASUREMENT .......................................................................................................... 49  

IV. CONCLUSION ....................................................................................................................... 60  
  4.1 RESULTS OF BEZIER SURFACE GENERATION .................................................................... 60  
  4.2 POTENTIAL FUTURE RESEARCH ....................................................................................... 62
APPENDIX A .......................................................................................................................... 64

MATLAB® CODE FOR BEZIER SURFACE GENERATION ALGORITHM ........................................... 65

REFERENCES ...................................................................................................................... 69
LIST OF FIGURES

Figure 1: Patella features........................................................................................................... 3
Figure 2: Cylindrical coordinate system..................................................................................... 3
Figure 3: First order Bezier parametric equation ...................................................................... 7
Figure 4: Cubic Bezier blending functions.................................................................................. 8
Figure 5: Cubic Bezier curve and control points ....................................................................... 12
Figure 6: Changing a control point in P_x and P_y................................................................... 13
Figure 7: Parameterized surface in u and v............................................................................. 15
Figure 8: Control point mesh arranged in a level plane............................................................ 16
Figure 9: Control point mesh example....................................................................................... 18
Figure 10: Corresponding parametric surface for above control points..................................... 18
Figure 11: Parametric surface and control mesh ..................................................................... 19
Figure 12: Only the corner points are shared............................................................................. 20
Figure 13: Control mesh and surface before moving control point........................................... 21
Figure 14: Control mesh after moving a control point P_{11} to -0.5 in z ................................. 21
Figure 15: Control mesh after moving a control point P_{11} to -2.0 in z ................................. 22
Figure 16: Convex control point mesh..................................................................................... 23
Figure 17: First Bezier Cubic Surface Q_1 ................................................................................. 24
Figure 18 Second Bezier Cubic Surface Q_2 ............................................................................. 24
Figure 19: Bezier control point mesh P and R of joined surfaces ............................................ 25
Figure 20: Joined Bezier parametric surfaces of Q_1 and Q_2 .................................................. 25
Figure 21: Initial raw data of the patella................................................................................... 32
Figure 22: Initial patella data points top view

Figure 23: Single 4x4 control point mesh of the patella U is 5x4

Figure 24: Single parametric patch with original data can control points

Figure 25: Single 4x4 control point mesh U is 9x4

Figure 26: Location of single patch of the patella control points

Figure 27: Single layer of the patella control points

Figure 28: Single layer of the patella control points and parametric surface

Figure 29: Peaks control points

Figure 30: Peaks control points and parametric surface

Figure 31: Generation of the control points for a sphere

Figure 32: Generation of the control points and parametric surface

Figure 33: Exceeding bounds of the data

Figure 34: Control point mesh for the entire patella front

Figure 35: Bezier parameterized surface patches entire patella front

Figure 36: Bezier control point mesh of entire patella top view

Figure 37: Bezier surface parameterization of entire patella top view

Figure 38: Affine transform of control points rotation about z

Figure 39: 2x scale in z

Figure 40: Patella scaled globally by 0.5

Figure 41: Patella contour parametric vs. original data (cm)

Figure 42: Patella contours top view parametric vs. raw data (cm)

Figure 43: Patella contours front view parametric vs. raw data (cm)

Figure 44: Patella contour xy plane 2 cm in z
Figure 45: *Patella contour yz plane 0 cm in x* ................................................................. 52

Figure 46: *Two patches modeled as one* ........................................................................ 54

Figure 47: *Peaks experiment* ........................................................................................ 56

Figure 48: *Sphere experiment* ....................................................................................... 57

Figure 49: *Patella experiment(x,y, and z in cm)* .............................................................. 58
LIST OF TABLES

Table 1: Error for single patch above................................................................. 36
Table 2: Percent error of the bounding box....................................................... 53
Table 3: Average and standard error for xy plane for crosssections in height z ....... 53
Table 4: Average and standard error for yz plan for cross sections in width .......... 53
Table 5: Error for two patches modeled as one.................................................. 54
Table 6: Error for two patches to two patches ................................................... 55
Table 7: Error for two patches to four patches................................................... 55
Table 8: Peaks experiment.............................................................................. 56
Table 9: Sphere experiment............................................................................ 57
Table 10: Patella experiment............................................................................ 58
PREFACE

This research develops a technique for the generation of the control points of Bezier surfaces. An algorithm for this technique is developed and is applied to the case study of the patella.

This approach is to use a pseudo-inverse technique to find the control points for a given set of 3-d raw graphic data which could be obtained via a 3-d scanning device. The control points are the defining points of the Bezier surface and lend information as to what the shape of the surface will be. The developed algorithm is developed to test the affects of the pseudo-inverse technique using the patella as a case study. The error will be calculated to measure the effects of surface approximation of the raw data.

The surface generating algorithm in Matlab code is applicable to many types of surfaces and is available in the Appendix of this thesis.
ACKNOWLEDGMENT

I would like to thank everyone who enabled me to consider and complete the research included in this thesis. I would like to thank all my undergraduate professors for presenting me with the knowledge I obtained, which was needed to follow through with my graduate studies. Also, my graduate professors for the opportunity to further increase my understanding of the Electrical Engineering field, which I am very motivated to be a part of. A special thanks goes out to Dr. Xue whose patience and understanding in answering my many questions greatly enhanced my ability to overcome obstacles, so I could continuously pursue the knowledge I dreamed of obtaining. I would also like to thank my technical leader at work Dr. Keith Jones for his advice on the non-technical aspects of thesis study that I sometimes overlooked. Next, I would like to thank ASC/ENM and ASC/XRE for funding most of my education while pursuing my Masters Degree in Electrical Engineering. Lastly but not least, I would like to thank my Mom whose accumulated hours of moral support gave me the energy to fulfill most of my educational goals.
I. INTRODUCTION

1.1 BACKGROUND

The design of curves, surfaces, and solids is and continues to be a complex and difficult problem, which was greatly and rapidly enhanced by the advent of the computer to apply numerical analysis to these types of problems. Mathematics can be considered to be the background in computer graphics, since it was invented first and the numerical methods involving 1s and 0s are eventually behind every digital computer graphic technique. Increases in computer performance have steadily made the parametric representation of physical objects found in the real world commonplace today for modeling, simulation, media, and obviously video games. Parametric representations are often chosen over explicit and implicit representations, since these two have deficiencies such as it being difficult to represent vertical lines in the explicit form, or having to test many points in order to find and equation in the implicit form [1]. Two clever engineers, Pierre Bezier and coincidently Paul de Casteljau invented Bezier curves [2][3]. Bezier curves are not called de Casteljau curves today because, although de Casteljau may have invented the idea first, but the company he worked for, Citroen, was very secretive about his work [3]. Nevertheless, his name is synonymous with Bezier curves today [4]. Parametric curves are defined by a set of discrete points called control points and basis functions which produce the shape of the curve. The technique developed by Bezier indirectly and intuitively allows for the specification and control of these parameters. Linear algebra is another mathematical tool that lends itself well to computer graphics
techniques, since computer memory is able to greatly enhance the speed at which affine matrix and vector operations are completed. This results in computer modeling and simulation that once could only be done by thought or real experiment alone.

Since the patella (knee cap) is used as a case study, special mention to the background as to why it was chosen and the features of its shape are given. The patella was selected for studying its shape because of its applicability to prosthesis design. The model of the joint at the knee at first glance seems to be a simple hinge, but upon further inspection the knee uses no pivot or pin for its construction [5]. The connection above the knee (AK) and below the knee (BK) is made by cruciate (crossing) ligaments which allow the knee to pivot. The patella is special in this case because it acts as a pivot of a lever and stays in place during movement at the knee. The patella is often considered to be at the center of the knee joint. This property of the patella frequently allows for its center to be used as a landmark or reference point in the design of leg prosthesis. In the cases needing an AK prosthesis the location where the patella would normally be located is determined by a virtual point in space. The features that make up the shape of the patella are illustrated below in Figure 1: Patella features. The left and right views are shown as seen from the front or anterior side, and the entire anterior region is considered to be convex from this vantage point. The articular surfaces are concave in order to properly fit the femur. There is also a slight concave shape that matches the tibia also, which is shown below the articular region of the patella. The anterior region is considered to be convex in these views.
1.2 APPROACH

Using the given shape information of the patella and information from a given set of coordinate points, the initial patella surface raw data is mapped using cylindrical coordinates. This coordinate system was chosen since most limbs are cylindrical in shape which fits in with the theme of prosthesis design for the knee. The cylindrical coordinates will be right handed (counter clockwise), and represented by the ordered triple \((r, \theta, z)\) as in Figure 2 [6].

**Figure 2: Cylindrical coordinate system**
The angle theta (θ) will be evenly spaced for reasons explained in the following sections and will be in degrees. The radius from the origin r in the xy-plane and the directed distance z from xy-plane will be measured in either centimeters or millimeters. Once this raw data is mapped in cylindrical coordinates, it is converted into right handed Cartesian coordinate system so that mathematical operations can be performed on the 3-d coordinate points (x,y,z) using MATLAB®.

Bezier curve generation in 2-d is investigated in order to understand and illustrate the characteristics of the 3-d Bezier surfaces. The 3-d Bezier surface patch will be generalized from these 2-d curves, since they share some of the same properties. This will be done by describing the control points for a surface mesh plus using the Bezier basis functions to create a parameterized surface. In order to test continuity across the boundary, two 3-d Bezier surface patches will then be joined together to form a single surface.

The initial raw data of the patella will be plotted in order to demonstrate the original size and shape. A reference frame will be drawn around the patella in order to observe the scaling and other affine transforms. The center of gravity and the geometric center of the patella will be found in order to produce a landmark for the knee center.

Next, the inverse process of determining the control points from a set of raw data as opposed to the given process from control points to parametric shape tested earlier. The pseudo inverse matrix technique will be applied in order to find the control points which correspond to the raw data by first looking at a single control point mesh of the patella. A single parameterized patch according to the derived control point positions should resemble the original data in shape and size. The pseudo inverse technique, the control
point mesh, and the resulting parameterized surface will be examined further by creating an entire row of control point meshes. A row being the height of the control points joined adjacently left to right in the xy-plane for 360° of the patella surface.

Once this is completed an algorithm to map both closed and open parameterized surfaces in their entirety will be developed. The purpose of the algorithm is to utilize the pseudo-inverse to find the corresponding control points, connect adjacent control points by position, and adjust to the boundaries of the raw data points. Control points will be derived from the surface point data in x,y, and z using MATLAB® given the original surface points for the given “peaks” and “sphere” shapes. These surfaces will be utilized in order to test the complete algorithm for needed improvements.

Affine transforms of the control points describing the entire 3-d model of the patella will be tested. Rotation, skewing, and scaling will be performed using the matrices described in the following section. Rotation will be performed in order to align the 3-d objects in relation to the coordinate system correctly. Skewing certain control points enables the patella shape to be changed in order to more closely match certain unique features. Scaling will be important in order to adjust the 3-d surfaces for various sized changes, while keeping the shape consistent. The resolution resulting from scaling will remain the same by increasing the number of parameters describing the surface accordingly.

Lastly, the error will be measured in order to get a quantitative magnitude of the differences between the $u \in [0,1]$ original raw data and the parameterized surfaces.
II. LITERATURE REVIEW

2.1 2-D BEZIER CURVES

Bezier curves in 2 dimensions are parametric polynomial curves. Bezier curves are an approximation that were first defined from geometric consideration [7], and later were shown to be equivalent to the Bernstein basis or polynomial approximation function shown below in equation Eq(1) [8]. Where $P_i$ are the control points which define the curve, $u$ is the parameter from 0 to 1, and $B_{i,n}(u)$ is the Bernstein basis or blending function Eq(2). In this equation $i$ is the $n$th order basis function, and $n$ is the highest degree of the polynomial.

\[ Q(u) = \sum_{i=0}^{n} P_i B_{i,n}(u) , u \in [0,1] \quad \text{Eq}(1) \]

\[ B_{i,n}(u) = \binom{n}{i} u^i (1-u) , \text{ where } \binom{n}{i} = \frac{n!}{i!(n-i)!} , u \in [0,1] \quad \text{Eq}(2) \]

Since Bezier curves were first defined geometrically, the illustrations below were created in order to portray them in a more intuitive form. Figure 3 shows the first order Bezier parametric equation, which in this case is a straight line segment.
There are two control points $P_i$, which are labeled as $P_0$ and $P_1$ ($n=0$ and $n=1$). The parameter $u$ starts at 0.0 on the left and ends at 1.0 on the right. The order of the polynomial is one described by $n$ being equal to 1. The mathematical form of this Bezier curve (line segment) is shown below in Eq(3). Notice that the segment and the control polygon occupy the same space, this only occurs for the first order instance.

$$Q(u) = P_0 + (P_1 - P_0)u = (1-u)P_0 + uP_1, \ u \in [0,1]$$  \hspace{1cm} \text{Eq(3)}$$

Looking at this parametric equation in terms of $u$ when $u=0.0$, $Q(0.0)=P_0$ the left endpoint in Figure 3. With $u=0.5$, $Q(0.5)=0.5P_0+0.5P_1$, and lastly with $u=1.0$, $Q(1.0)=P_1$ the right end point. Here $u$ is continuous from 0 to 1 in theory, but using numerical analysis techniques require that an incremental change or $u_i+\Delta_i$ be considered within an adequate resolution.

Eq(4) shows the third order mathematical form of the Bezier curve derived from Eq(1) and Eq(2) above. This cubic equation is described by the four control points $P0$, $P1$, $P2$, and $P3$. The parameter $u$ again varies from 0 to 1, with $n=3$ which is also the highest order of the parametric curve in this instance.
Figure 4: Cubic Bezier blending functions

The cubic polynomial in Eq(4) is given by

\[ Q(u) = B_{0,3} P_0 + B_{1,3} P_1 + B_{2,3} P_2 + B_{3,3} P_3, \quad u \in [0,1] \]  

\[ Q(u) = (1-u)^3 P_0 + 3u(1-u)^2 P_1 + 3u^2 (1-u) P_2 + u^3 P_3, \quad u \in [0,1] \]  

Figure 4 shows the four Bezier blending basis function that are needed to describe the cubic polynomial in Eq(4). Each blending function is a cubic polynomial, and the sum of the four blending functions at any value in \( u \) is equal to 1. Comparing this with Eq(5) one can see the influence each control point has on the final curve [9]. As an example, at \( u \) near zero \( P_0 \) multiplied by \( B_{0,3} = (1-u)^3 \) clearly has the greatest influence, with \( B_{0,3} \) being equal to the total of the constant sum of one for the value of \( u=0 \) [9]. The control points \( P_1 \) and \( P_2 \) have their greatest influence at their peaks of \( u=1/3 \) and \( u=2/3 \) accordingly [9]. \( P_3 \) has its greatest influence at \( u=1 \), where all other cubic polynomials are equal to 0. The
use of these blending functions has properties associated with them outlined below. This is a consequence of Bezier basis being the same as the Bernstein basis [10].

- The basis functions are real
- The degree of the polynomial defining the curve segment is one less than the number of control points defining the polygon (open meaning the first and last points are not joined).
- The curve follows the shape of the defining polygon formed by connecting the control points.
- The first and last points on the curve coincide with the first and last points of the defining polygon.
- The tangent vectors at the first and last points are respectively equal to the tangent of the first and last polygon spans.
- The curve is contained within the convex hull of the enclosed defined closed polygon.
- The curve does not oscillate about any straight line more often than the defining polygon. This is known as the variation diminishing property.
- Bezier curves can be mapped using affine transforms, or Bezier curves are affine invariant. You need only to transform the control points and then compute the new curve.

Some of these properties are also inferred by the de Casteljau algorithm and lend intuitive interpretation to the geometric shape of the Bezier curves and associated polygons [11]. These properties discussed for further explanation to lend insight as to why they are important.

Affine transforms can be applied to Bezier curves because they are affine invariant [11]. This essentially signifies that if the affine mapping is applied to either the control points $P_i$, or to the points on the curve itself $Q(u)$ the result is the same. This allows for applying for example a rotation to only a few control points, and generating the parameterization of a great number of parameterized points on the curve. This can be
performed more quickly than the converse operation of doing an affine transform on all the parameterized points on the curve, and then determining the few control points to describe the shape.

The invariance under affine parameter transformation property defines the transition in parameterization from \([a_1, b_1]\) to \([a_2, b_2]\). It applies to Bezier basis functions since the parameter \(u\) was defined as \(u \in [0, 1]\). This is important yet not necessary [11], since parametric equations for curves are not unique because the parameters can be defined over any arbitrary interval from \(\bar{u}_{\text{min}} < \bar{u}_1 < \bar{u}_2 < \ldots < \bar{u}_{m-1} = \bar{u}_{\text{max}}\) [12]. It is important to normalize the interval \([\bar{u}_{\text{min}}, \bar{u}_{\text{max}}]\) to \([u_{\text{min}}, u_{\text{max}}]\)=[0,1] since it provides consistent and convenient unit weighting from the beginning to the end of the curve [13]. If the interval wasn’t normalized in this way it would be confusing. The reason being that contribution of each value of the parameter from \([\bar{u}_{\text{min}}, \bar{u}_{\text{max}}]\) would produce a result that would have an effect on the basis function values. The effect could be predicted, but it would be an added difficulty in determining the control points each time the values of the parameters changed as in the examples of Eq(3), Eq(4), and Eq(5). The normalization to determine the parameters \(u\) and \((1-u)\) are shown in the following Eq(6) and Eq(7). Here \(\bar{u}\) is the value starting from the left end of the interval \(\bar{u}_i\) to the right end of the interval \(\bar{u}_{i+1}\). The delta \(\Delta_i\) from one value of \(\bar{u}\) to the next for numerical analysis purposes in plotting the curve is chosen to be sufficiently small, and usually but not necessarily equal. For example \(\Delta_i=0.01\) for an entire interval \(\bar{u}_i \leq \bar{u} \leq \bar{u}_{i+1}\). The knot values \(u_i\) and \(u_{i+1}\) are determined in relation to the interval of the given raw data, which also depends on how much accuracy is desired. Generally the more closely spaced the knot values are in relation to the given data the more accurate the parameterized curve. Curvature also
needs to be considered for the knot spacing. Only two points are needed to define a straight line, but the greater the curvature the more knot values are needed to define the curve accurately. Another consideration is placing a knot at a sharp edge, since Bezier has positional continuity careful placement of knots at this edge can model this condition correctly. Knot spacing for random data is particularly difficult for Bezier, but it can be done since Bezier is an approximation to a curve. Less knots per data point has a tendency to smooth the curve, where more knots per raw data typically shows more of the original features (sharp edges, curvature, noise, etc.).

\[
u = \left( \frac{\bar{u} - \bar{u}_i}{\bar{u}_{i+1} - \bar{u}_i} \right) \quad \text{Eq}(6)
\]

\[
(1 - u) = \left( \frac{\bar{u}_{i+1} - \bar{u}}{\bar{u}_{i+1} - \bar{u}_i} \right) \quad \text{Eq}(7)
\]

The convex hull property is a result of the convex barycentric combination of control points. This is another reason why it is important to normalize the parameter from \([\bar{u}_{\min}, \bar{u}_{\max}]\) as in \textbf{Eq}(6) and \textbf{Eq}(7). If this wasn’t done the sum of Bezier Blending functions shown in Figure 4 would not be equal to 1. Barycentric combinations are defined as coefficients that must sum to be equal to 1 and are also non-negative, so it is necessary for this to be true to produce convex combinations. The convex hull property can be explained in terms of a peg board analogy. If pegs are placed at random on a peg board, the convex hull would be the equivalent to stretching a rubber band around the furthest points on the peg board and letting it snap. The resulting polygon would be the convex hull. The convex hull helps to intuitively determine the shape of the curve. It also
serves as a way to check that the generated parametric curve is correctly defined as a Bezier curve. For example, if this curve exceeds the boundary of the convex hull then something is creating an error in the algorithm used to generate control points and/or parameterized curve.

End point interpolation is a property that is due to the endpoints being defined by \( P_0 \) and \( P_3 \) at the ends of the parameterization at \( u=0 \) and \( u=1 \), and can be seen by observing the values Figure 4.

The last property associated with the de Casteljau algorithm is that the control polygon seems to follow the shape of the control polygon [14]. This provides an intuitive idea of what the shape of the curve will be and makes it possible to define the shape of the curve through iteration.

Now that the properties of cubic Bezier curves have been listed and commented on, it is illustrative to see some of these properties through an example.

**Figure 5:** *Cubic Bezier curve and control points*
In Figure 5 a classic Bezier curve example with the control points arranged in a polygon the shape of a square with $P_0=(0,0)$, $P_1=(0,1)$, $P_2(1,1)$, and $P_3=(1,0)$ is shown. Here the properties described above are in agreement with example in Figure 6 below. $P_x$ and $P_y$ show the effect of moving the control points by increments of $\Delta P = 0.2$ in both the x and y directions. Here the effect of moving the control point can be seen for two separate polygon shapes $P_0P_1P_2P_3$ and $P_0P_4P_6P_5$. From this initial condition $P_2$ for the first polygon and $P_6$ for the second are successfully moved from $P_6$ to $P_{15}$ in the direction of the arrow. The effect on the curvature on the smaller polygon is greater than the case of the larger polygon. The successive change in one point of the two polygons in this example also affects the entire curve. This illuminates the fact that Bezier curves do not have local control. At first this may seem to have a negative connotation, but sometimes this property is useful especially when dealing with probabilistic data. Also, when it is important to have local control the weight of one particular points location can be increased by placing successive points at that same location. This can provide a very close approximation to having local control.

**Figure 6: Changing a control point in $P_x$ and $P_y$**
Exact amount that the parametric curve changes shape can be seen by adding the $\Delta P$ (amount of change of a control point) to $P_2$ to Eq(5) and then subtracting $Q_1$ and $Q_2$. Where $Q_1$ is the orginal equation and $Q_2$ is $Q_1$ with the amount of change. Which results in Eq(8) immediately below.

$$\Delta Q(u) = 3u^2(1-u)\Delta_p$$

Eq(8)

2.2 3-D BEZIER SURFACES

As in the 2-d case Bezier surfaces the Bernstein basis is used for the surface blending functions. Since this is used as the basis many properties of the surface can be determined and are listed here [15] [16].

- The degree of the surface in a single parametric direction is one less than the number of defining polygon vertices in that direction.
- The continuity of the surface in a single parametric direction is two less than the number of defining polygons in that direction.
- The surface intuitively follows the general shape of the defining control polygon.
- Only the four corner points of the control polygon and the parameterized surface must be shared, coincident, or interpolated.
- The surface is contained within the convex hull of the defining polygon mesh control points.
- “The surface does not exhibit the variation diminishing property. The variation diminishing property for bi-variant surfaces is both undefined and unknown. [15].”
- The surface is invariant when affine transforms are applied to either the control points or parameterized surface.
- The basis functions are nonnegative.
The basis functions sum to 1 for all \( u \) and \( v \).

The parameter \( v \) is introduced since the curve is generated from a two dimensional Cartesian product Bezier mesh. A sketch of a parameterized surface in \( u \) and \( v \) can be seen in Figure 7. Notice that one of the corner control points \( P_{00} \) can be seen as incident with the parameterized surface, this is only true for the corner points of the mesh. Figure 8 shows a control point mesh for a single patch arranged in a level plane for illustrative purposes. This patch consists of four points in the \( v \) direction and four in the \( u \) direction for a total of sixteen points for a cubic patch. The corner points \( P_{00}, P_{03}, P_{33}, \) and \( P_{30} \) are the only points which must be coincident to the parametric surface. All the other points can be moved to control the shape of the 3-d surface. Other geometric properties of this control point mesh are the tangent vectors in \( u \) and \( v \), and the twist vector shown as in equations Eq(9), Eq(10), and Eq(11) respectively [17].

Figure 7: Parameterized surface in \( u \) and \( v \)
Figure 8: Control point mesh arranged in a level plane

\[ Q_u(0,0) = 3(P_{01} - P_{00}) \]  \hspace{1cm} Eq(9)

\[ Q_v(0,0) = 3(P_{10} - P_{00}) \]  \hspace{1cm} Eq(10)

\[ Q_{uv}(0,0) = 9(P_{00} - P_{01} - P_{10} + P_{11}) \]  \hspace{1cm} Eq(11)

The Bezier surface is generated in much the same way as the Bezier curves in the 2-d case. A cubic Bezier patch is defined as shown in Eq(12) [18], where \( Q(u,v) \) is the parameterized surface, \( P_{ij} \) is the control point mesh, \( B_i \) and \( B_j \) are the Bezier basis functions, and \( i \) and \( j \) are the respective indices. Given the control points \( P_{ij} \) and the Bezier basis functions in \( u \) and \( v \) a surface as in can be generated.

\[ Q(u,v) = \sum_{i=0}^{3} \sum_{j=0}^{3} P_{ij} B_i(u)B_j(v) \]  \hspace{1cm} Eq(12)

It is often convenient to show this in matrix form as shown below in Eq(13). Here the parameters \( u \) and \( v \) are vectors and the coefficients of the Bezier basis functions are
arranged in a matrix. The control points $P$ are also arranged in a 4x4 matrix, which coincides with the control point mesh in Figure 8.

$$Q(u,v) = [u^3 \ u^2 \ u \ 1][BPB^T]$$

Eq(13)

$$\begin{bmatrix}
-1 & 3 & -3 & 1 \\
3 & -6 & 3 & 0 \\
-3 & 3 & 0 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}$$

Eq(14)

$$P = \begin{bmatrix}
P_{00} & P_{01} & P_{02} & P_{03} \\
P_{10} & P_{11} & P_{12} & P_{13} \\
P_{20} & P_{21} & P_{22} & P_{23} \\
P_{30} & P_{31} & P_{32} & P_{33}
\end{bmatrix}$$

Eq(15)

Examples of generating a parametric surface given a set of control points in $P_x$, $P_y$, and $P_z$ are illustrated in the following figures. The matrices for the 4x4 control points are shown below as in Eq(15). Comparing this with Figure 9: Control point mesh example, the control points $P_x$, $P_y$, and $P_z$ form the shape of corresponding bi-parametric surface shown in Figure 10. The properties of the Bezier surface described in the beginning of this section can be observed as well. Figure 9 illuminates the count of the first property listed with four control points in any one direction for a cubic Bezier surface patch.

$$P_x = \begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
2 & 2 & 2 & 2 \\
3 & 3 & 3 & 3
\end{bmatrix}, \ P_y = \begin{bmatrix}
0 & 1 & 2 & 3 \\
0 & 1 & 2 & 3 \\
0 & 1 & 2 & 3 \\
0 & 1 & 2 & 3
\end{bmatrix}, \ P_z = \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 2 & 3 & 1 \\
1 & 2 & 3 & 1 \\
1 & 1 & 1 & 1
\end{bmatrix}$$
**Figure 9:** Control point mesh example

**Figure 10:** Corresponding parametric surface for above control points
The control mesh and bi-parametric surface are superimposed on Figure 11 in order to gain a more intuitive understanding of how the control point mesh influence the scaling and the shape of the parametric surface. Here it can clearly be seen that the parametric surface fits within the convex hull of the control points, and that the shape is characteristic of the polygon mesh of control points that define it. One item in the list of characteristics that may be misleading here is that the four corner points of the control polygon and the parameterized surface must be are shared, coincident, or interpolated. The fact that other perimeter points are coincident in this case is a result from the tangent vectors having a slope of zero, which is not always true.
A counter example to Figure 11 is shown in Figure 12, which helps to demonstrate the property that only the four corners of a Bezier surface patch are coincident in every case. This is analogous with the 2-d instance where only the corner ends of the segment are coincident with the parametric curve. Here only the four corner points are labeled in the sixteen control point mesh described in Figure 8. The adjacency of the arrangement remains the same however. The case in which the control point mesh is arranged in a flat plane as in Figure 8 is also analogous to the straight line example in 2-d. The analogy being that it is the only situation in which the all of the control points and the corresponding parameterized points will lay on the same plane, yet still be within a complex hull of control points.
Figure 13: Control mesh and surface before moving control point

Figure 14: Control mesh after moving a control point $P_{11}$ to -0.5 in z

$$P_x = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \end{bmatrix} \quad P_y = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{bmatrix} \quad P_z = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -0.5 & 3 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$
Figure 14 above demonstrates the effect moving a control point has on the corresponding parametric surface compared to the original example in Figure 13. The point $P_{11}$ in the control matrix $P_z$ is and in Figure 14 are highlighted for comparison.

Figure 15: Control mesh after moving a control point $P_{11}$ to -2.0 in $z$

\[
P_z = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \end{bmatrix} \quad P_y = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 \end{bmatrix} \quad P_z = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -2 & 3 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}
\]

Comparing Figures 13, 14 and 15 above demonstrate the effect of moving a control point on the control point mesh to an even greater extent. The moved control point is highlighted in Figure 15 and in $P_z$ above. In this example the parametric surface is affected once again, and a convex region on the parameterized surface observing from the top is formed by moving this control point.
Figure 16: Convex control point mesh

\[
\begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
2 & 2 & 2 & 2 \\
3 & 3 & 3 & 3
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & 1 & 2 & 3 \\
0 & 1 & 2 & 3 \\
0 & 1 & 2 & 3
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & -2 & -3 & 1 \\
1 & -2 & -3 & 1 \\
1 & 1 & 1 & 1
\end{bmatrix}
\]

Figure 16 show the inner points of \(P_z\) inverted with respect to Figure 11, and the entire parametric surface is convex when observing from above as showing the effect of moving multiple control points and that the surface can be inverted.

Joining Bezier surface patches is often a matter of maintaining first-order continuity [19]. This involves both positional and tangential continuity across the boundary. The conditions require that the control points and the characteristic polygon are shared between adjoining patches [19]. The conditions involving zero-order continuity or positional continuity for a cubic polynomial patch are shown below, with \(P\) being the first control point mesh and \(R\) representing the second [19].
\begin{align*}
P_{33} &= R_{03} & \text{Eq}(16) \\
P_{32} &= R_{02} & \text{Eq}(17) \\
P_{31} &= R_{01} & \text{Eq}(18) \\
P_{30} &= R_{00} & \text{Eq}(19)
\end{align*}

**Figure 17:** First Bezier Cubic Surface \( Q_1 \)

**Figure 18** Second Bezier Cubic Surface \( Q_2 \)
Figure 19: Bezier control point mesh $P$ and $R$ of joined surfaces

![Bezier control point mesh](image)

Figure 20: Joined Bezier parametric surfaces of $Q_1$ and $Q_2$

![Joined Bezier parametric surfaces](image)
Cubic Bezier surfaces are shown above joined according to Eq(16) to Eq(19) using only positional continuity of the control points. Here both the control points are shared across the boundary and the parameterized points of $Q_1$ and $Q_2$ are shared as a result. This allows for the maintaining of positional continuity between the two patches. The sharp edge shows that tangential continuity does not exist in this case. Many techniques have been developed that are less restrictive than the first order continuity conditions. Positional continuity can be used alone for sharp edges such as these without gradient continuity. However, it is still necessary that the tangent vectors at the edges meeting at a corner must be coplanar [20].

There are various ways to numerically enforce continuity between Bezier patches depending on the actual shape of the object and the need for directional tangents to be equal. Many algorithms and techniques have been developed in order to join two patches together. This is done because not all data can be presented in the same way. In this research effort all the control point meshes are rectangular, as is the predominate case, but some shapes are better served by triangular or circular patches. It is also sometime useful to use uniform and non-uniform subdivision of patches, but this increases the potential for holes between the patches resulting from the approximation of a patch boundary thru a straight line [21].

Another method is to increase the size of the control mesh by increase the order of the Bezier basis function used. All these techniques have limitations, and are sometimes difficult to implement. No one technique has been proven to cover all possible instances of joining approximated or interpolated raw data the best. Plus, human beings have philosophical knowledge of boundaries at infinity, limits that approach an infinitesimal amount, and other mathematical concepts that deal with boundary conditions that are difficult to instill artificially into a computer.
2.3 THE GEOMETRIC CENTER AND THE CENTER OF MASS

Since the case study for this involves using the patella as a reference point for prosthetic knee, it is important to find the center of mass of the patella in order to find a more precise location for a landmark point. This point can be used for measurements of displacement around the knee, and as a reference for the affine and other transforms. The geometric center is found by simply taking the maximum and minimum value in each axis x, y, and z and then dividing by two to get the average or geometric center. The centroid or center of mass is found by summing entire surface points of the raw data, as well as the theoretical points within the patella which make it a geometric solid volume.

\[
\bar{x} = \frac{\sum_{i=1}^{n_x} x_i}{n_x} \quad \text{Eq}(20)
\]

\[
\bar{y} = \frac{\sum_{i=1}^{n_y} y_i}{n_y} \quad \text{Eq}(21)
\]

\[
\bar{z} = \frac{\sum_{i=1}^{n_z} z_i}{n_z} \quad \text{Eq}(22)
\]

2.4 PSEUDO INVERSE MATRIX

The pseudo-inverse matrix provides the least mean squares error solution to a system of linear equations. It is also known as the Moore-Penrose matrix inverse [22]. A system of equations \( Ax = y \) can be solved for \( x \) by using the inverse where \( A \) is invertible \( A^{-1} \).
This is done by \( x=Ax^{-1}y \). This requires that for matrix \( A \) to be invertible the number of rows \( m \) must be equal to the number of columns \( n \), or the number of equations is equal to the number of unknowns. If the number of equations \( m \) is greater than the number of unknowns \( n \) or \( m>n \) it is considered to be overdetermined. An overdetermined system is usually, but not always inconsistent. An example where an overdetermined system would be consistent or having an exact solution would be three straight lines which happen to pass through the same point. Since this is unlikely to happen every time a system is overdetermined, the pseudo-inverse of a matrix \( A \) is used in order to find the least mean squares solution and is denoted by \( A^+ \). This is described as \( A^+=(A^TA)^{-1}A^T \). The matrix \( A \) has a pseudo-inverse if \((A^TA)^{-1}\) exists [23]. This is true since multiplying \( A^+ \) by \( A \) and rearranging the equation yields \(((A^TA)^{-1}A^TA)=(A^TA)^{-1}(A^TA)=1 \) [23]. The least squares solution analogous to the determined inverse solution described earlier is shown below in equations Eq(23-25). Multiplying both sides of the overdetermined system by the pseudo-inverse conveniently yields the least mean square solution \( x_m \).

\[
\begin{align*}
Ax &= y \quad \text{Eq(23)} \\
x &= [(A^TA)^{-1}A^T]y \quad \text{Eq(24)} \\
x_m &= A^+y \quad \text{Eq(25)}
\end{align*}
\]

2.5 MEASURING ERROR

Since the Bezier surfaces regenerated by using a few of control points estimated from the raw data are an approximation to the original patches, the least mean squares is calculated to find for the estimation error results from this. In the case for the
overdetermined solution finding control points using the pseudo-inverse matrix produces the best solution in regards to minimizing the average error [24]. Here in Eq(26) the average error is \( e_{av} \), the parameterized points are \( Q(u,v) \), the original raw data points are represented by \( p_i \), and \( N \) is the upper limit of the indexed number of data.

\[
e_{av} = \sum_{i=0}^{N} \| Q(u,v) - p_i \|^2
\]

Eq(26)

The standard error \( e_s \) provides the standard deviation of the average error for the methods used to generate Bezier surfaces in this research [25]. This is given in Eq(27) in the form of a formula that can be directly translated to code.

\[
e_s = \sqrt{ \frac{ \sum e_{av}^2 - \frac{1}{N} (\sum e_{av})^2 }{ N - 1 } }
\]

Eq(27)

Contour plots can also be used for 3-d surfaces in order to give an intuitive look at the amount of error involved. The differences in the 3-d patches in 2-d contours will be done for left, right, top, front, and back in order to get the all the six perspective in a bounding box.

2.6 AFFINE TRANSFORMS

Affine transformations are defined in a geometric setting as precisely the functions that map lines to lines [26]. Most transformations which rotate, scaled, or translate an object in computer graphics are affine maps [27]. As stated earlier, the Bezier surface is
invariant when affine transforms are applied to either the control points or parameterized surface. The associated transformation matrices are defined below. Eq(28) shows the portioning of the transformation matrix with $R_{3x3}$ being the rotation submatrix, $p_{3x1}$ the position vector for translation, $f_{1x3}$ for perspective transformation, and $s_{1x1}$ for scaling.

\[
T = \begin{bmatrix}
R_{3x3} & p_{3x1} \\
\end{bmatrix}
\]

Eq(28)

\[
T_{x,\alpha} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & \cos(\alpha) & -\sin(\alpha) & 0 \\
0 & \sin(\alpha) & \cos(\alpha) & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Eq(29)

\[
T_{y,\phi} = \begin{bmatrix}
\cos(\phi) & 0 & \sin(\phi) & 0 \\
0 & 1 & 0 & 0 \\
-\sin(\phi) & 0 & \cos(\phi) & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Eq(30)

\[
T_{z,\theta} = \begin{bmatrix}
\cos(\theta) & -\sin(\theta) & 0 & 0 \\
\sin(\theta) & \cos(\theta) & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Eq(31)

\[
T_{\text{tran}} = \begin{bmatrix}
1 & 0 & 0 & dx \\
0 & 1 & 0 & dy \\
0 & 0 & 1 & dz \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Eq(32)

\[
T_{\text{scale}} = \begin{bmatrix}
a & 0 & 0 & 0 \\
0 & b & 0 & 0 \\
0 & 0 & c & 0 \\
0 & 0 & 0 & s \\
\end{bmatrix}
\]

Eq(33)
Equation Eq(29) represents a rotation about the x-axis where the angle $\alpha$ is measured by rotating the original Z-Y plane to the new Zn-Yn plane counter clockwise when looking at the origin from a point on the positive X axis [28]. Equations Eq(30) and Eq(31) are the associated rotations about the y and z-axis respectively. Eq(32) represents the translation matrix in which the coordinates are translated parallel to a reference coordinate system by the amount $dx$, $dy$, and $dz$. Eq(33) is the scaling transformation matrix, where a, b, and c scale the points x, y, and z respectively, and the scale factor s is a global scaling factor for x, y, and z.
III. DEVELOPMENT

3.1 INITIAL RAW DATA

The raw data of the patella describe the initial size and shape of the features described in the introduction. The initial data of the patella can be seen in Figure 21 below, where the bounding box, centroid, and geometric center can be also seen in centimeters. The bounding box encapsulates the patella from $x_{min}$ to $x_{max}$, $y_{min}$ to $y_{max}$, and $z_{min}$ to $z_{max}$. Using equations Eq(20) thru Eq(22) the centroid was found and is labeled as $\bar{x}$, $\bar{y}$, and $\bar{z}$. The geometric center is designated as $x_c$, $y_c$, and $z_c$, which is equal to half the distances of each axis of the bounding box or for example $(x_{min} + x_{max})/2$.

**Figure 21: Initial raw data of the patella**
The top view of the patella looking down from the z axis can be seen in Figure 22 below. Here it can be seen that the data was originally in cylindrical coordinates as described in Figure 2, with the independent variable angle $\theta$ in equally spaced in $5^\circ$ increments. Although it is not necessary for Bezier surfaces to be mapped with the independent variable in equal increments, it is convenient to do so since it provides uniform spacing that can be studied more conveniently. Also, uniform b-splines require this to be true, so it could become necessary if the basis functions were to change. This data will be converted into Cartesian coordinates, so that the common commands and function in MATLAB® can be used. This is a change in mapping in coordinate system, but the location of the points remains the same.

Figure 22: Initial patella data points top view
3.2 THE PSEUDO INVERSE TECHNIQUE

The process described in section 2.2 Eq(13) through Eq(15) is the given forward process, which describes finding a parameterized surface \(Q(u,v)\) with the given control points as in Eq(15). The reverse process is to use the Bezier surface to describe the shapes of a set of given graphic data points. This is done by setting the parameters in a matrix \(U\) and \(V\), the pseudo inverse of a matrix could be applied to the forward process given in Eq(13) to find the control points \(P\). This is derived in Eq(35-40) below where \(P_s\) are the original graphic data points shown in Figure 21, \(U^+\) and \(V^+\) are the pseudo inverse matrices, \(B\) is the Bezier basis matrix coefficients given in equation 14. \(P\) is the resulting control point matrix found by Eq(39). The pseudo inverses of matrices of parameters \(U\) and \(V\) must be taken instead of the normal inverse because it is likely that the parameterization produces a matrix size that is greater than the 4x4 Bezier basis matrix as the example shown in Eq(34) for matrix \(U\) and \(V\) for \(u\) and \(v\) with six parameters to match the dimension of the original graphic data points.

\[
U = \begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & u_1 & u_1 & 1 \\
1 & u_2 & u_2 & 1 \\
1 & u_3 & u_3 & 1 \\
1 & u_4 & u_4 & 1 \\
1 & u_5 & u_5 & 1 
\end{bmatrix}, \quad V = \begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & v_1 & v_1 & 1 \\
1 & v_2 & v_2 & 1 \\
1 & v_3 & v_3 & 1 \\
1 & v_4 & v_4 & 1 \\
1 & v_5 & v_5 & 1 
\end{bmatrix}
\]

\[
P_s = UBPB^TV^T
\]

\[
U^TP_sV = U^TUBPB^TV^TV
\]

\[
[(U^TU)^{-1}U^T]P_s[V(V^TV)^{-1}]^T = [(U^TU)^{-1}(U^TU)]BPB^T[(V^TV)(V^TV)^{-1}]
\]

\[
U^+P_s(V^+)^T = BPB^T
\]

\[
P = [(B^{-1})(U^+)]P_s[(V^+)^T(B^{-1})^T]
\]
3.3 SINGLE CONTROL POINT MESH OF THE PATELLA

Using the technique described in by Eq(39), a single patch of the patella is created. Here the control point mesh was created from the original data $P_s$, and a parameterized surface $Q(u,v)$ was drawn using these control points $P$. This parameterized surface resembles the shape of a single bottom layer patch from the original surface points $P_s$ as intended. The number of sampled points $P_s$ is a 5x5 matrix. The parameter values of the cubic polynomial $U$ and $V$ are 5x4 matrices.

**Figure 23:** Single 4x4 control point mesh of the patella $U$ is 5x4
Figure 24: Single parametric patch with original data can control points

Table 1: Error for single patch above

<table>
<thead>
<tr>
<th>Point</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>QX</th>
<th>QY</th>
<th>QZ</th>
<th>real error x</th>
<th>real error y</th>
<th>real error z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q00</td>
<td>18</td>
<td>0</td>
<td>10</td>
<td>18.18</td>
<td>0.00377</td>
<td>9.941</td>
<td>-0.18</td>
<td>0.003767</td>
<td>0.059</td>
</tr>
<tr>
<td>Q01</td>
<td>16.94</td>
<td>1.482</td>
<td>10</td>
<td>16.99</td>
<td>1.48</td>
<td>9.941</td>
<td>-0.05</td>
<td>0.002</td>
<td>0.059</td>
</tr>
<tr>
<td>Q02</td>
<td>15.46</td>
<td>2.726</td>
<td>10</td>
<td>15.71</td>
<td>2.7868</td>
<td>9.941</td>
<td>-0.25</td>
<td>-0.0608</td>
<td>0.059</td>
</tr>
<tr>
<td>Q03</td>
<td>14.49</td>
<td>3.882</td>
<td>10</td>
<td>14.41</td>
<td>3.87</td>
<td>9.941</td>
<td>0.08</td>
<td>0.012</td>
<td>0.059</td>
</tr>
<tr>
<td>Q04</td>
<td>13.16</td>
<td>4.788</td>
<td>10</td>
<td>13.17</td>
<td>4.87</td>
<td>9.941</td>
<td>-0.01</td>
<td>0.001</td>
<td>0.059</td>
</tr>
<tr>
<td>Q10</td>
<td>15.38</td>
<td>7.5</td>
<td>14.7</td>
<td>0.00509</td>
<td>7.734</td>
<td>0.6</td>
<td>-0.00509</td>
<td>-0.234</td>
<td></td>
</tr>
<tr>
<td>Q11</td>
<td>15.24</td>
<td>7.5</td>
<td>14.51</td>
<td>1.261</td>
<td>7.734</td>
<td>0.73</td>
<td>0.072</td>
<td>-0.234</td>
<td></td>
</tr>
<tr>
<td>Q12</td>
<td>14.08</td>
<td>7.5</td>
<td>13.87</td>
<td>2.437</td>
<td>7.734</td>
<td>0.21</td>
<td>0.046</td>
<td>-0.234</td>
<td></td>
</tr>
<tr>
<td>Q13</td>
<td>13.23</td>
<td>7.5</td>
<td>13.04</td>
<td>3.514</td>
<td>7.734</td>
<td>0.19</td>
<td>0.046</td>
<td>-0.234</td>
<td></td>
</tr>
<tr>
<td>Q14</td>
<td>12.22</td>
<td>7.5</td>
<td>12.31</td>
<td>4.471</td>
<td>7.734</td>
<td>-0.09</td>
<td>-0.025</td>
<td>-0.234</td>
<td></td>
</tr>
<tr>
<td>Q20</td>
<td>10</td>
<td>0</td>
<td>5</td>
<td>10.97</td>
<td>1.55E-6</td>
<td>4.649</td>
<td>-0.97</td>
<td>-1.545E-06</td>
<td>0.351</td>
</tr>
<tr>
<td>Q21</td>
<td>10.46</td>
<td>5.9151</td>
<td>11.3</td>
<td>0.9928</td>
<td>4.649</td>
<td>-0.84</td>
<td>-0.0777</td>
<td>0.351</td>
<td></td>
</tr>
<tr>
<td>Q22</td>
<td>10.14</td>
<td>1.789</td>
<td>5</td>
<td>10.87</td>
<td>1.903</td>
<td>4.649</td>
<td>-0.73</td>
<td>-0.114</td>
<td>0.351</td>
</tr>
<tr>
<td>Q23</td>
<td>10.14</td>
<td>2.718</td>
<td>5</td>
<td>10.16</td>
<td>2.735</td>
<td>4.649</td>
<td>-0.02</td>
<td>-0.017</td>
<td>0.351</td>
</tr>
<tr>
<td>Q24</td>
<td>9.679</td>
<td>3.523</td>
<td>5</td>
<td>9.611</td>
<td>3.493</td>
<td>4.649</td>
<td>0.068</td>
<td>0.03</td>
<td>0.351</td>
</tr>
<tr>
<td>Q30</td>
<td>7</td>
<td>1.5</td>
<td>6.349</td>
<td>0.00431</td>
<td>1.734</td>
<td>0.651</td>
<td>0.004311</td>
<td>-0.234</td>
<td></td>
</tr>
<tr>
<td>Q31</td>
<td>7.272</td>
<td>0.6362</td>
<td>1.5</td>
<td>6.726</td>
<td>6.017</td>
<td>1.734</td>
<td>0.546</td>
<td>0.0345</td>
<td>-0.234</td>
</tr>
<tr>
<td>Q32</td>
<td>6.894</td>
<td>1.216</td>
<td>1.5</td>
<td>6.391</td>
<td>1.113</td>
<td>1.734</td>
<td>0.503</td>
<td>0.103</td>
<td>-0.234</td>
</tr>
<tr>
<td>Q33</td>
<td>5.796</td>
<td>1.553</td>
<td>1.5</td>
<td>5.797</td>
<td>1.558</td>
<td>1.734</td>
<td>-0.001</td>
<td>-0.005</td>
<td>-0.234</td>
</tr>
<tr>
<td>Q34</td>
<td>5.356</td>
<td>1.95</td>
<td>1.5</td>
<td>5.398</td>
<td>1.965</td>
<td>1.734</td>
<td>-0.042</td>
<td>-0.015</td>
<td>-0.234</td>
</tr>
</tbody>
</table>
**Figure 24** and **Table 1** above shows the error for a single patch of the patella. $Q_x$, $Q_y$, and $Q_z$ are the parameterized surface patches and $x$, $y$, and $z$ are the original data coordinates in millimeters. The average error $e_{avg}$ was found to be 0.33805 mm for $x$, 0.3371 mm for $y$, and 0.2195 mm for $z$. The standard error $e_s$ was found to be 0.321236 mm for $x$, 0.03516251 mm for $y$, and 0.106959 mm for $z$.

**Figure 25:** *Single 4x4 control point mesh $U$ is 9x4*

![Image of control point mesh](image_url)

**Figure 25** shows an example where the number of points sampled is a 4x4 matrix. The pseudo inverse technique must be applied because $u$ and $v$ changed from $[0.0, 0.25, 0.50, 0.75, 1]$ to $[0, 0.10, 0.20, \ldots, 0.80, 0.90, 1]$ resulting a change from five parameters to eleven. Each row in matrix $U$ and $V$ is being evaluated for a single parameter $u$ and $v$. The control point mesh is 4x4, and the parameterized polynomials $U$ and $V$ are therefore 9 rows and 4 columns.
\[
Q(u,v) \approx P_s = UBPB^T V^T
\]
\textbf{Eq(40)}

In order to clarify this, \(P_s\) is the original data in \textbf{Eq(39)}, but if solved for the forward case as in \textbf{Eq(13)} it should be approximately equal to the parameterized surface \(Q(u,v)\) from \textbf{Eq(40)} and \textbf{Figure 24}.

3.4 SINGLE CONTROL POINT MESH LAYER OF THE PATELLA

\textbf{Figure 26} shows the location of the control point mesh in red and the parameterized surface in light grey in \textbf{Figures 23 to 25} in relation to the original raw data of the entire patella. A single control point mesh layer is found for 360° about the z axis. A single layer being 4x4 control point meshes connected adjacent left to right for 360° as in \textbf{Figure 27}.

\textbf{Figure 26: Location of single patch of the patella control points}
The process of modeling the patella is followed through by creating a single layer of joined patches. The positional continuity is preserved by applying Eq(16-19) on the bottom layer of Figure 26. The resulting control point mesh is shown in Figure 27.

**Figure 27:** Single layer of the patella control points

![Single layer of the patella control points](image)

**Figure 28:** Single layer of the patella control points and parametric surface

![Single layer of the patella control points and parametric surface](image)
Since the pseudo-inverse matrix produces the least mean squares approximation the boundaries at the ends of the control point mesh are not exactly equal. The boundaries are then joined together by averaging and setting them to be equal for positional continuity. The control points here form 15 4x4 meshes to produce the parameterized surface in Figure 28. In this example there are 132 control points, 891 parametric points, and 73 rows and 10 columns or 730 original data points. The control point to original data ratio is 0.18 and the original data points were sampled 10 rows by 10 columns at a time.

3.5 BEZIER SURFACE GENERATING ALGORITHM

Since the entire patella consists of many data points, it would be more difficult to model than the previous examples. At this point it was decided that an algorithm for the MATLAB® code should be developed in order to understand and improve the code as it was written. An outline of the algorithm is included here for convenience, and the entire MATLAB® code for generating these surfaces is included in the Appendix A.

1.) Collect initial raw data
2.) Determine Parameter size for U and V
3.) Determine sampling patch size from surface patches Ps (original data)
4.) Use Pseudo Inverse technique to create control points P
5.) Create the corresponding parametric surface
6.) Use positional continuity and averaging to join adjacent patches left right until the end of the first row
7.) Start the next column and use positional continuity and averaging to join the patches together by row
8.) Test to see if the end samples exceed the total number of data points if so add the remainder for fit, if not the process is complete

Using this algorithm, test examples of the MATLAB® peaks and sphere where tested.

**Figure 29:** Peaks control points

**Figure 30:** Peaks control points and parametric surface
Figure 31: Generation of the control points for a sphere

Figure 32: Generation of the control points and parametric surface
These surfaces help in the debugging process of the code generated to create the patella since they consist of fewer points. The sphere resembled the patella in this case because they are both closed surfaces. The peaks surface in Figure 30 has 2,401 original data points, 784 control points, and 3,527 parametric points. The size of the original data was 8 rows by 8 columns, and the control point to original data ratio is 0.33. The sphere in Figure 32 has 148 control points, 441 original points, and 1,001 parametric points. The step size was 10 original data points by 10 points, and the number of control points to original data point ratio is 0.34.

Figure 33 is an illustration of why tests in the algorithm at step 8 was included in the code. When the step size which helps to determine the number of raw data points collected exceeds the number of amount of data available, MATLAB® generates an error because the matrix exceeds the bounds of the array and the resulting plot is open at the top of the surface. This was solved by adding the remainder to the matrix indices until the indices plus remainder is equal to the data bounds.

Figure 33: Exceeding bounds of the data
3.6 ENTIRE BEZIER SURFACE OF THE PATELLA

The final results of applying the algorithm and modeling the entire patella can be seen in the following Figures 34 to 37 there were 1533 original data points.

**Figure 34:** Control point mesh for the entire patella front

![Control point mesh for the entire patella front](image)

**Figure 35:** Bezier parameterized surface patches entire patella front

![Bezier parameterized surface patches entire patella front](image)
Figure 36: Bezier control point mesh of entire patella top view

Figure 37: Bezier surface parameterization of entire patella top view
In the above Figures the original data points were selected in groups of 5x5 and were mapped with a total of 1,204 control points for a control point to data points ratio of 0.7854.

3.7 AFFINE TRANSFORMATION

Affine transformations are useful once the control point mesh is found. The transforms can be applied quickly since there are less control points than parameterized points or data points. The control points can be reduced by selecting a larger sample patch size or step size looking at the code. For example, a sample patch size of 5x5 would have 25 original data points for every 16 control point patches for a control point to original data ratio of 0.64, but a sample patch size of original data that was 10x10 would produce a ratio of around 0.16. This doesn’t always occur however because of the way that the control patches are matched at the beginning and ending boundaries. This occurs since there is round off error between the total number of data points and the control point patch size as described from Figure 33.

Figure 38 below shows an affine transform of the control points using Eq(31), rotation about the z axis. In this case the patella control points are rotated by 45° and the parameterized surface is then recalculated to align back with the convex hull of the control point mesh. Compared with the corresponding plot in Figure 37, this demonstrates that the properties described in sections 2.2 3-d Bezier surfaces, and 2.6 Affine transforms hold true.
Once this done, the parameterized surface needs to be redrawn to match the new rotated position of the control points.

Other transformation using equations using \textbf{Eq.(28-33)} can be applied to the control points as well. Scaling is particularly useful, since the patella varies in size from person to person. Affine transformations can also be applied to groups of control point adjust the shape of the patella within certain region.

\textbf{Figure 38} shows the effect of rotation of 45° and 2x scale in z using affine transformation of the original data. Although this can be done is can be done more quickly by using the control points. Also, buy using the control points the parameterization can be increased or decreased as in \textbf{Figures 19} and \textbf{20}. This allows for the resolution to remain consistent when applying scaling.
Figure 39: 2x scale in z

Figure 40: Patella scaled globally by 0.5
3.8 ERROR MEASUREMENT

Since Bezier surfaces are an approximation and the pseudo inverse provides the least mean squares solution, there should predictably be some amount of error. Although Bezier surfaces can be just as accurate as uniform B-spline surfaces or NURBS (non-uniform b-spline surfaces) the fact that they do not have local control (moving one control point affects the entire curve) makes it more difficult to achieve this level of accuracy. The more Bezier surfaces are understood however, the more the advantages of using them become apparent. This includes the shape being similar to the control mesh that defines it.

Figure 41: *Patella contour parametric vs. original data (cm)*
Figure 41 illustrates the various contour lines looking down the z axis onto the xy plane. On the right is the parameterized surface and on the left are the original data points. At first it may appear that the shape is consistent, but there is quite a bit of error in some of the contours. This figure however, illustrates another point. Even though the original data and the Bezier parametric surfaces share the exact same coordinate system it does not guarantee the matrices containing the data will be the same size, or share the exact same data points between the two plots. This is true even if an exact solution was reached. Since the Bezier surface is parametric, the equation can be solved for a two particular coordinates of the original data to subtract from the third coordinate to get the error. For example find the solution $Q_x(u,v) \approx x$ and $Q_y(u,v) \approx y$, once the exact $u$ and $v$ parameters are found the approximate error is equal to $Q_z(u,v)-z$. This is no trivial task however, especially if matrices $U$ and $V$ are large.

Figures 42 and Figures 43 show other contours which match. The dash dot line being the original data, and the solid line being the contours of the parameterized surface. Using these figures the average error and the standard error can be found. Figure 42 shows contours looking down the z-axis from above, and Figure 43 show the contours looking from the front of the patella down the x-axis. These contours were made from a patella with an 8x8 sample data patch size, so the approximation shows observable error although the shape is preserved correctly. The number of control points is 480, and the number raw data points is again 1533 giving a control point to raw data point ratio of 0.31. The number of parameterized points can vary, but in this instance it is 2,169. For example, $u$ and $v$ can simply be [0.0,0.5,1.0] which results in 183 parameterized points.
Figure 42: Patella contours top view parametric vs. raw data (cm)

Figure 43: Patella contours front view parametric vs. raw data (cm)
Figure 44: Patella contour xy plane 2 cm in z

Figure 45: Patella contour yz plane 0 cm in x

Figure 44 shows a single contour in the xy-plane, which is 2 cm above the bottom (z=0cm) of the patella. Here the amount of error can be more clearly seen, with the blue
or dashed line being the original data, and green solid line the parameterized contour data. Figure 45 is an example of a single layer in the yz plane with \( x = 0 \)cm. The difference in these contours are used to measure the average and standard error.

<table>
<thead>
<tr>
<th>Axis</th>
<th>( P_{\text{min}} ) (cm)</th>
<th>( P_{\text{max}} ) (cm)</th>
<th>( P_{\text{min}} ) (cm)</th>
<th>( P_{\text{max}} ) (cm)</th>
<th>% error ( \text{min} )</th>
<th>% error ( \text{max} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>-2.0500</td>
<td>2.300</td>
<td>-2.0524</td>
<td>2.3026</td>
<td>-0.1171</td>
<td>-0.1130</td>
</tr>
<tr>
<td>y</td>
<td>-1.3098</td>
<td>1.330</td>
<td>-1.3201</td>
<td>1.3475</td>
<td>-0.7864</td>
<td>-1.3158</td>
</tr>
<tr>
<td>z</td>
<td>0.0100</td>
<td>4.620</td>
<td>0.0041</td>
<td>4.6168</td>
<td>-59.00</td>
<td>0.06926</td>
</tr>
</tbody>
</table>

Table 1 shows the percent error in the bounding box axis between Figure 17 and the upper and lower limits of the parameterized surface.

### Table 3: Average and standard error for xy plane for cross sections in height \( z \)

<table>
<thead>
<tr>
<th>xy-plane (z cm)</th>
<th>Average error x</th>
<th>Average error y</th>
<th>Standard error x</th>
<th>Standard error y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.125</td>
<td>0.779</td>
<td>0.531</td>
<td>0.732</td>
</tr>
<tr>
<td>0.5</td>
<td>0.152</td>
<td>1.221</td>
<td>0.541</td>
<td>0.993</td>
</tr>
<tr>
<td>1.0</td>
<td>0.041</td>
<td>1.008</td>
<td>0.055</td>
<td>0.995</td>
</tr>
<tr>
<td>1.5</td>
<td>0.274</td>
<td>0.936</td>
<td>0.457</td>
<td>0.998</td>
</tr>
<tr>
<td>2.0</td>
<td>1.05</td>
<td>1.007</td>
<td>0.995</td>
<td>0.996</td>
</tr>
<tr>
<td>2.5</td>
<td>0.134</td>
<td>0.549</td>
<td>0.536</td>
<td>0.467</td>
</tr>
<tr>
<td>3.0</td>
<td>2.00</td>
<td>0.856</td>
<td>0.998</td>
<td>0.831</td>
</tr>
<tr>
<td>3.5</td>
<td>1.225</td>
<td>0.978</td>
<td>0.997</td>
<td>0.992</td>
</tr>
<tr>
<td>4.0</td>
<td>0.210</td>
<td>0.550</td>
<td>0.468</td>
<td>0.488</td>
</tr>
</tbody>
</table>

### Table 4: Average and standard error for yz plane for cross sections in width

<table>
<thead>
<tr>
<th>yz-plane (z cm)</th>
<th>Average error y</th>
<th>Average error z</th>
<th>Standard error y</th>
<th>Standard error z</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2.00</td>
<td>0.128</td>
<td>0.840</td>
<td>0.334</td>
<td>0.992</td>
</tr>
<tr>
<td>-1.56</td>
<td>1.130</td>
<td>0.841</td>
<td>0.992</td>
<td>0.993</td>
</tr>
<tr>
<td>-1.11</td>
<td>1.06</td>
<td>0.840</td>
<td>0.995</td>
<td>0.503</td>
</tr>
<tr>
<td>-0.67</td>
<td>0.730</td>
<td>0.631</td>
<td>0.720</td>
<td>0.125</td>
</tr>
<tr>
<td>-0.22</td>
<td>0.200</td>
<td>0.221</td>
<td>0.433</td>
<td>0.701</td>
</tr>
<tr>
<td>0.22</td>
<td>0.430</td>
<td>0.993</td>
<td>0.220</td>
<td>0.993</td>
</tr>
<tr>
<td>1.11</td>
<td>0.315</td>
<td>0.735</td>
<td>0.455</td>
<td>0.642</td>
</tr>
<tr>
<td>1.56</td>
<td>0.980</td>
<td>0.217</td>
<td>0.992</td>
<td>0.321</td>
</tr>
<tr>
<td>2.00</td>
<td>1.200</td>
<td>1.250</td>
<td>0.331</td>
<td>0.093</td>
</tr>
</tbody>
</table>
Tables 2 and 3 demonstrate the average $e_a$ and standard error $e_s$ using the formulas in Eq(26) and Eq(27). This was done for nine contour levels in both the xy plane and yz plane as in Figures 37 and 38.

**Figure 46:** Two patches modeled as one

![Image of two patches modeled as one](image)

**Figure 46:** Two patches modeled as one above shows the results of modeling the earlier Figures 19 and 20, which consisted of two control patches, into now being modeled with only one control point patch for this test case of control point reduction. The mean error resulting from this reduction in the following table is the average error between the original and generated surface parameters.

<table>
<thead>
<tr>
<th>Data Sampled</th>
<th>Mean Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 36x64</td>
<td>0.04462</td>
</tr>
<tr>
<td>2, 18x36</td>
<td>0.04711</td>
</tr>
<tr>
<td>4, 9x18</td>
<td>0.04949</td>
</tr>
</tbody>
</table>
The data samples are the data points which were taken for each original data point, every other data point, and every fourth data point. The total sample size of the original data is shown adjacent to this example in a 9x18 patch 162 original data points were used as sampled data surface points. The mean error in each of the following cases is the distance from the original data point to the generated parameterized point divided by the number of samples. The control point error in Table 6 is calculated since the generated control points correspond one to one to the original points.

<table>
<thead>
<tr>
<th>Data Sampled</th>
<th>Mean Error</th>
<th>Control Point Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 36x64</td>
<td>4.9743x10^{-29}</td>
<td>0.0243</td>
</tr>
<tr>
<td>2, 18x36</td>
<td>6.2402x10^{-29}</td>
<td>0.0976</td>
</tr>
<tr>
<td>4, 9x18</td>
<td>0.003900</td>
<td>0.4614</td>
</tr>
</tbody>
</table>

Table 6 shows the mean error and control point error for generating the equivalent 8x4 control point mesh as the original. The error for the two patch to two patch example is much smaller than the earlier case of two patches to one as would be expected. The control point error can be measured directly from the original control points in Table 6, since there is a one to one correspondence between the original number of control points and the regenerated points.

<table>
<thead>
<tr>
<th>Data Sampled</th>
<th>Mean error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 36x64</td>
<td>0.006223</td>
</tr>
<tr>
<td>2, 18x36</td>
<td>2.5592x10^{-29}</td>
</tr>
<tr>
<td>4, 9x18</td>
<td>0.0038997</td>
</tr>
</tbody>
</table>
Table 7 shows the results of increasing the number of control point patches from two to four over the surface of the Figures 19 and 20. Here the error is still lower in the one patch case, but is less than that of simply trying to regenerating the original control points.

Figure 47: Peaks experiment
Mean Error 0.00087227

Table 8: Peaks experiment

<table>
<thead>
<tr>
<th>Data Sampled</th>
<th>Control points</th>
<th>Mean error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 50x50</td>
<td>12x12</td>
<td>0.0459</td>
</tr>
<tr>
<td>2, 25x25</td>
<td>12x12</td>
<td>0.1116</td>
</tr>
<tr>
<td>4, 13x13</td>
<td>12x12</td>
<td>0.0029</td>
</tr>
<tr>
<td>1, 50x50</td>
<td>20x20</td>
<td>0.00087</td>
</tr>
<tr>
<td>2, 25x25</td>
<td>20x20</td>
<td>0.00025</td>
</tr>
</tbody>
</table>

Figure 47 along with Table 8 demonstrate the same experiment with the peaks function in MATLAB® with a total of 50x50 original data points in x, y, and z. Here the number of control points were increased, and the sampling was done for every point, every other point, and every fourth point of the original data. The lower mean error for the higher
number of data samples can be explained since in these cases there was no rounding of the number of data samples relative to the original data, so the patches lined up more closely.

**Figure 48: Sphere experiment**

![Sphere experiment](image)

**Table 9: Sphere experiment**

<table>
<thead>
<tr>
<th>Data Sampled</th>
<th>Control points</th>
<th>Mean error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 100x100</td>
<td>4x4</td>
<td>0.021454</td>
</tr>
<tr>
<td>2, 50x50</td>
<td>4x4</td>
<td>0.1116</td>
</tr>
<tr>
<td>4, 25x25</td>
<td>4x4</td>
<td>0.0029</td>
</tr>
<tr>
<td>1, 100x100</td>
<td>8x8</td>
<td>0.00015251</td>
</tr>
<tr>
<td>2, 50x50</td>
<td>8x8</td>
<td>0.00014735</td>
</tr>
<tr>
<td>4, 25x25</td>
<td>8x8</td>
<td>0.00013494</td>
</tr>
<tr>
<td>1, 100x100</td>
<td>40x40</td>
<td>3.475x10^{-10}</td>
</tr>
<tr>
<td>2, 50x50</td>
<td>40x40</td>
<td>8.9061x10^{-11}</td>
</tr>
</tbody>
</table>

**Figure 48** and **Table 9** display the error of the experiment involving the sphere. This figure shows an extreme example of modeling a 100x100 point sphere in x, y, and z.
using only four control points, with the mean error being significantly greater than the cases involving a greater number of control points.

**Figure 49: Patella experiment (x, y, and z in cm)**

Mean Error 0.003659

![Patella experiment graph](image)

<table>
<thead>
<tr>
<th>Data Sampled</th>
<th>Control Points</th>
<th>Mean error (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 21x72</td>
<td>4x4</td>
<td>0.078755</td>
</tr>
<tr>
<td>4, 5x18</td>
<td>4x16</td>
<td>0.014909</td>
</tr>
<tr>
<td>2, 11x36</td>
<td>8x16</td>
<td>0.0049758</td>
</tr>
<tr>
<td>1, 21x72</td>
<td>16x16</td>
<td>0.0036691</td>
</tr>
<tr>
<td>2, 11x36</td>
<td>8x36</td>
<td>0.003858</td>
</tr>
<tr>
<td>1, 21x72</td>
<td>12x48</td>
<td>0.0049646</td>
</tr>
<tr>
<td>1, 21x72</td>
<td>20x72</td>
<td>0.0002749</td>
</tr>
</tbody>
</table>

**Table 10: Patella experiment**

**Figure 49** and **Table 10** show the results of various control points describing the patella with an original raw data set of 21x73 data points in x, y, and z. It was found that the minimum control point to data ratio while still preserving a low error was for the case of 16x16 control. This has a control point to raw data ratio of 256:1533 or 0.167. The 20x72
control point example has the lowest error, but the number of control points is almost equal to the amount of raw data with a control point to raw data ratio of 0.9393.
IV. CONCLUSION

4.1 RESULTS OF BEZIER SURFACE GENERATION

In this thesis the patella was modeled as a case study to re-create the shape of the patella using Bezier basis functions to create a control point mesh. This was investigated by first looking at the 2-d instances of using the Bezier basis functions along with the given control points to find the shape, which characteristically described by the control point polygon. The other properties of the 2-d Bezier basis functions where also investigated in order to gain a better understanding of the 3-dimensional case. The Bezier basis functions and given control points were then used to create a mesh of control points in 3-d. The boundary conditions at various patches of the Bezier surfaces where then determined for positional continuity, and two patches where joined together. The raw data of the case study of the patella was then used to find the centroid, geometric center, and bounding box.

The pseudo-inverse matrix of the parametric functions of u and v where then used to find the least means square solution to the problem of having a given set of raw data in x, y, and z find the corresponding control points which describe the shape. This was tested successfully on a single patch of the patella, and again using a greater degree of parameterization.

Once this was accomplished joining the control points using positional continuity as with joining the parameterized points was tested. Positional continuity by sharing neighboring control points proved to be a successful method of joining parameterized surfaces.
An algorithm was developed in order to understand the more complex cases of joining surface patches which consisted of many layers and sometimes a closed shell. This algorithm was then successfully tested with the MATLAB® peaks and sphere generated points, and the shape was recreated with marginal error.

The algorithm was then tested on the patella, and it was found that the indices which controlled the positional continuity and step size could exceed the matrix dimensions which contained the points. This was solved by shifting the back indices to the end of the matrix dimensions, and calculating the final layer.

Affine transforms were then applied to the control points as well as the original points. Global scaling as well as rotation about the axis were tested, and found to be consistent with the Bezier surface properties of affine transformation. A scale factor to maintain the parameterization resolution was used for the corresponding affine scaling transformation. This allowed the shaped to increase or decrease while maintaining the same resolution.

Lastly, the error was measured and it was found that the maximum values of height, length, and width which made up the bounding box were accurate for the parameterized surface in terms of absolute error. Contours of the patella were then generated to measure the average error and standard deviation of equal layers of the patella raw data and parametric surface of the patella in the xy and yz planes. The curves were found to have values which were verified to acceptably describe the original patella shape.
4.2 POTENTIAL FUTURE RESEARCH

The potential for future research projects are immense. Bezier surfaces are traditionally used for design work, and although they can be very accurate the fact that they do not have local control (moving one control point moves the entire curve) means that it takes more effort to get the same level of accuracy as uniform B-spline surfaces or non-uniform B-splines (NURBS). Bezier curves do have the advantage that they can describe complex and random data more easily. For this reason this technique may be applied to feature detection of complex shapes such as facial expressions.

Landmarks could also be added to the control point meshes in order to describe and vary the shape of the Bezier surface.

It can be shown that Bezier curves are a special case of b-spline curves, so creating different aspects of the transformations involved between the basis functions could be useful.

Studying how the introducing control points in order find the minimum number of control points for a needed accuracy given a certain amount of curvature. As an example, extreme cases of a finite flat plane or sharp edge could be used. A flat plane would need only 16 control points no matter what its dimensions would be. Also, sharp edges could be found using the limits of curvature, and Bezier end points could be placed there to preserve sharp edges where this occurred.

The foundations and applications of grid generation could also be explored in order to gain and understanding of conformal mapping and adaptive grids.
Further study into the applications of 3-d computer graphics using the technique described in this thesis or other towards the applications of prosthetic design.
MATLAB® code for Bezier Surface generation algorithm

clear all;

% steps sample size
s = 4

% [ Pxp  Pyp Pzp ] = peaks(50)
[Px, Py, Pz] = sphere(100)
% Pxp = x;
% Pyp = y;
% Pzp = z;

% input data
x = Px(1:s:100,1:s:100)
y = Py(1:s:100,1:s:100)
z = Pz(1:s:100,1:s:100)

% initialize parameterized surface matrices
Pxuv = 0;
Pyuv = 0;
Pzuv = 0;

% x= PxuvT
% y= PyuvT
% z= PzuvT

% initialize row and column index of parameteric surface to one
psri = 1;
psci = 1;

% determine number of rows of the given data
maxrows = size(x,1)
% determine the number of columns of given data
maxclmn = size(x,2)

% Bezier Basis function
B = [-1 3 -3 1; 3 -6 3 0; -3 3 0 0; 1 0 0 0;]

% sample size
% sub 4
smpsizer = 4;
smpsizec = 4;
% find the number of samples given the maximum rows and columns of the
% original data
rjc = floor(maxrows/smpsizer);
cic = floor(maxclmn/smpsizec);

% find the remainder of the entire data
remr = mod(maxrows,smnsizer);
remc = mod(maxclmn,smnsizec);

% test for boundary at the end of the entire matrix in both columns and
% rows, if exceeded increase sample size
for j1 = 1:rjc
    for i1 = 1:cic
        if j1 >= rjc
            mri = j1*smnsizer+remr;
            jc = (mri-smnsizer-remr+1):mri;
        end
        if i1 >= cic
            mci = i1*smnsizec+remc;
            ic = (mci-smnsizec-remc+1):mci;
        end
    end
end

% size scalar of raw input data
usize = size(jc,2);
size = size(ic,2);

% sampled points % commented out are affine transformation about z
Px = x(jc,ic);%.*0.707-y(jc,ic).*0.707;
Py = y(jc,ic);%.*0.707+x(jc,ic).*0.707;
Pz = z(jc,ic);

% corresponding parameterized interval to the samples above
u = (0:1/(usize-1):1)';
v = (0:1/(vsize-1):1)';

% Create the U and V cubic polynomial matrix for the sampled points
U = [u.^3 u.^2 u ones(length(u),1)];
V = [v.^3 v.^2 v ones(length(v),1)];
%Create the parameterization for u and v which define the resolution of the
%parameterized surface does not need to be equal to u and v
u2 = (0:1/(usize-1):1)';
v2 = (0:1/(vsize-1):1)';

% u2 = (0:1/20:1)';
% v2 = (0:1/20:1)'

% determine the size scalar for the generated parametric surface
u2size = size(u2,1);
v2size = size(v2,1);

% generate the matrices U and V for the psuedo inverse transformation
U2 = [u2.^3 u2.^2 u2 1:1]';
V2 = [v2.^3 v2.^2 v2 1:1]';

% control point matrix indices
cmri = j1*4;
cmci = i1*4;

$$$$ psuedo-inverse transformation to determine control points
$$$$

Pxc(cmri-3:cmri,cmci-3:cmci) =
((inv(B)*pinv(U))*Px*(pinv(V)'*inv(B)'));
Pyc(cmri-3:cmri,cmci-3:cmci) =
((inv(B)*pinv(U))*Py*(pinv(V)'*inv(B)'));
Pzc(cmri-3:cmri,cmci-3:cmci) =
((inv(B)*pinv(U))*Pz*(pinv(V)'*inv(B)'));

% parameterized surface point matrix indices
jcv = psci:psci+v2size-1
icu = psri:psri+u2size-1

$$$$ determine parameterized surface given derived control points
$$$$

Pxuv(icu,jcv) = U2*B*Pxc(cmri-3:cmri,cmci-3:cmci)*B'*V2';
Pyuv(icu,jcv) = U2*B*Pyc(cmri-3:cmri,cmci-3:cmci)*B'*V2';
Pzuv(icu,jcv) = U2*B*Pzc(cmri-3:cmri,cmci-3:cmci)*B'*V2';

% test to see if the furthest boundary of original data point is reached
% if so reset the parameterized surface column index to 1 and increase the
% row index by the size of u2
if il >= cic
    psci=1;
    psri = psri+u2size;
else
    psci=psci+v2size;
end % if il >= cic

end % for il
end % for j1

hold on;

% calculate the error
dx = Pxs(1:s:100,1:s:100)-Pxuv
dy = Pys(1:s:100,1:s:100)-Pyuv
dz = Pzs(1:s:100,1:s:100)-Pzuv

% hypotenuse of the error
h = sum(sum(dx.^2+dy.^2+dz.^2))

% mean error
N = size(Pxuv,1)*size(Pxuv,2);
me = h/N;

figure(1)
colormap('default')

% plot parameterized surface
surf(Pxuv,Pyuv,Pzuv);grid on; % title(['Mean Error ',num2str(me)])
xlabel('x');ylabel('y');zlabel('z');

% plot control points
plot3(Pxc,Pyc,Pzc,'-b.');grid off;
plot3(Pxc',Pyc',Pzc','-k.');%axis([-1.3 1.3 -1.3 1.3 -1.3 1.3]);view([45 57.8]);
% plot original data
% surf(x,y,z)
hold off
REFERENCES


[28] Kefu Xue, Ph. D, *Digital Image Processing Class Notes, Imaging Geometric and Camera Model*, EE715 Wright State University, pg. 2