Computer Science
Technical Report

Colorado State University

Fast Generation of NURBS Surfaces from Polygonal Mesh Models of Human Anatomy

Charles W. Anderson and Steward Crawford-Hines

February 9, 2000

Technical Report CS-99-101

Computer Science Department
Colorado State University
Fort Collins, CO 80523-1873

Phone: (970) 491-5792 Fax: (970) 491-2466
WWW: http://www.cs.colostate.edu

Abstract

Visible Productions, Inc., of Fort Collins, CO, produces 3-D human models that are recognized as some of the most accurate models in the world. Their models currently are based on meshes of 3-D triangles. Such meshes can be rendered as smooth surfaces by interpolating color values across a triangular mesh, but for a number of applications the smooth surface must be explicitly represented. Clients for Visible Productions' models have asked for surfaces defined by NURBS (Non-Uniform Rational B-Splines). This project developed and implemented algorithms for transforming polygonal meshes into NURBS. This requires a time-intensive, iterative optimization process. We investigated the use of neural networks to by-pass a large part of the optimization process.

1 Problem

Visible Productions, Ltd., of Fort Collins, CO, and other companies produce polygon-based, 3-D models of organic structures in humans and animals. There is already a large database of excellent 3-D anatomical models existing that was created at high expense. Figure 1 shows several of the many polygon-based models created by Visible Productions. These models are used primarily for educational uses in schools and in marketing of pharmaceuticals and medical devices. New applications for these models require that they be represented in a compact, mathematically organized format. These models need to be capable of easy deformation to demonstrate complicated biochemical and physiological functions: movement of living tissue, effects of drugs on tissues/organs, and simulation of anatomical structures interacting with surgical devices and diagnostic instruments.

One way to represent organic structures is as a volume of densities. This is a very intuitive representation, but volume models require much storage space. Most applications need only the surface of the structure, so explicitly representing the surface is much more storage-efficient than are volume models. Surface models can also more accurately represent organic structures. Surface models based on polygons are very useful at this time, because many graphics boards contain hardware for accelerating the rendering of polygons. Polygon-based surface models, although much smaller in size than volume-based models, are still very large and hard to manage. Relatively inexpensive software that can convert polygon models into mathematically efficient and compact

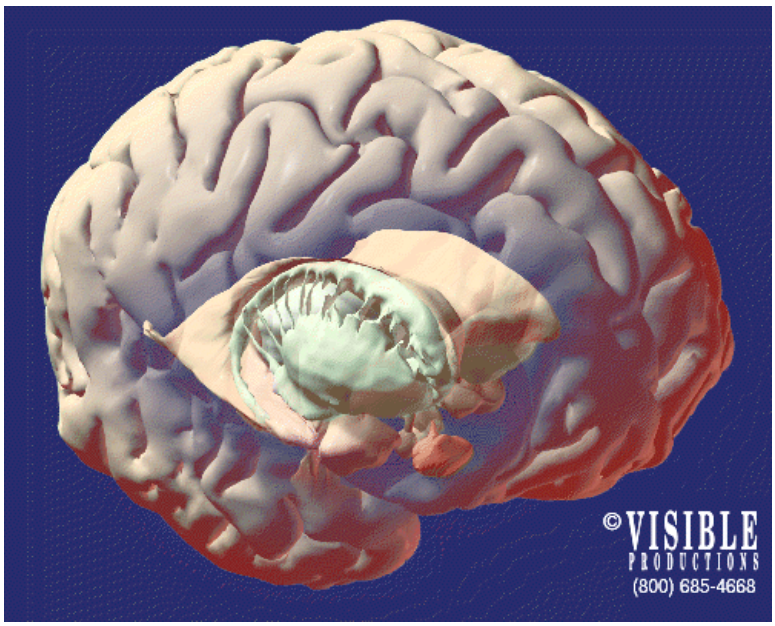

Figure 1: Polygon-based model produced by Visible Productions, Inc.

models in a fast and accurate manner would be of great value to any company that needs to manipulate 3-D models.

The ready availability of accurate polygon surface models of anatomical objects would have a great impact in various medical, educational, diagnostic, and clinical applications. As telemedicine becomes more important and the need to transfer anatomical data sets in real time over computer networks becomes more urgent, the requirement that the models be efficiently realized and compactly represented is critical. The ease of ready manipulation of the models in educational applications is important for students and patients in understanding the geometry and the pathologies involved. In diagnosis, the calculation of surface area and volumes (for example, the volume of brain ventricles in the cerebral cortex) is made computationally more efficient by a polygon-based or in general a surface-based model representation of the organ in question.

One way to represent easily-manipulated, smooth surfaces that is common in CAD (Computer-Aided Design) is NURBS (Non-Uniform Rational B-Splines) surfaces. A NURBS representation is a smooth parametric model for the surface in question, while retaining a compactness of representation which offers great economies in manipulation, storage, transmission, and uniformity over a wide variety of platforms and applications.

A number of companies, including Johnson and Johnson, US Surgical, and Boston Tech, have told Visible Productions that they would purchase their models if they were represented in NURBS. Any company that needs to manipulate Visible Productions' models or any other polygon models using standard CAD software, such as ProEngineer, must purchase the models as NURBS surfaces, the standard form accepted by CAD packages.

No software exists today that quickly and accurately generates NURBS surfaces for organic structures. Currently, only three software packages exist for generating NURBS from 3-D points. One is called Surfacer, by Imageware, Inc., and another is called Conversion, by Alias Wavefront. Both of these cost approximately $20,000 and are geared toward inorganic structures. They are very inefficient and inaccurate when working with models of organic structures, because there is a much finer grain of detail in organic structures than in the inorganic models for which these packages are normally used. A third product is MedCAD by Materialise USA, which suffers from some of the same deficiencies when applied to anatomical models. Visible Productions has evaluated the existing software by attempting to create NURBS models of some of Visible Productions' models. We found that the large computational time required to accurately approximate the polygon-based surfaces made these tools impractical. This is verified by the fact that a company named Zygote recently spent a full year to produce a fairly accurate NURBS model of an entire human skeleton.

Visible Productions has a unique opportunity to combine the mesh-to-NURBS conversion tool to be developed on this project with a new mesh-generation tool Visible Productions developed with a prior NSF SBIR Phase I award and currently being refined through a Phase II SBIR award. The prior project resulted in software that uses neural networks to assist skilled human tracers in identifying boundaries in 2-D slices. Neural networks are trained to model the decisions the humans make as they use a mouse to draw boundary lines. After a small number of boundary points are identified, the neural network is trained to also choose those points. Then, the neural network very quickly extends the boundary trace from the point at which the human paused. When the neural network makes a mistake, or is unable to confidently predict the next point, the human takes over and traces additional points. These points become additional examples on which the neural network is trained. Publications by Crawford-Hines and Anderson [3, 4, 5] describe this work in more detail.

2 Objective

The objective of this project is to develop software that converts 3-D polygon-mesh models of organic structures into NURBS surfaces that accurately approximate the polygon data and that accomplish this in a practical amount of time. For Phase I, our first objective is to implement and demonstrate

an algorithm for optimizing a NURBS surface for the approximation data generated by hand-tracing the contours of regions of interest in the Visible Human data set. Our second Phase I objective is to demonstrate a way in which neural networks can be applied to this problem to decrease the time required to optimize the fit of a NURBS surface to the data.

3 Background

NURBS have become a standard in computer-aided design (CAD), because they have the following properties [8]:

- NURBS provide local control, as do B-Splines. Moving one control point only affects the surface shape near the control point. Designing with local control is much more intuitive than designing with surfaces without local control, such as Bezier surfaces.
- They are invariant under scaling, translation, shear, and rotation, as are other smooth surface representations like Bezier and B-Spline surfaces. This means that these transformations need only be applied to the control points of the surface, not to every point on the surface. Unlike other representations, NURBS are also invariant under perspective transformations. This saves much time during rendering.
- NURBS surfaces can be used to exactly define quadric surfaces, such as spheres and ellipsoids, shapes that are common in organic structures. B-Splines can only approximate such surfaces and require many more control points to do so.

An example of a NURBS surface is shown in Figure 2. The mesh of control points are shown above the surface. Let $c_{i,j}$ be the mesh of control points, as i and j vary along the two dimensions of the mesh. To define a point on the NURBS surface, a weighted average of nearby control points is calculated. The weighting is specified by the B-Spline blending functions $N_{i,p}(u)$, where p is the order of the function, i indicates this is the i^{th} blending function, and it is a function of the parameter u that varies along one dimension of the control point mesh. With the additional weighting factor $w_{i,j}$, the equation for the point on the NURBS surface corresponding to parameter values u and v is

$$p(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}c_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}}.$$

The B-Spline blending functions are defined recursively as

$$\begin{aligned} N_{i,0}(t) &= \begin{cases} 1, & \text{if } t_i \le t < t_{i+1} \text{ and } t_i < t_{i+1}; \\ 0, & \text{otherwise,} \end{cases} \\ N_{i,p}(t) &= \frac{t - t_i}{t_{i+p} - t_i} N_{i,p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(t) \end{aligned}$$

The t_i's are the elements of the knot vectors. A common form for the knot vector of non-uniform B-Splines is for the first $p+1$ knots to be 0 and the last $p+1$ knots to be 1. The remaining knots must form a non-decreasing sequence of real numbers.

The tensor product expression for a point, $P(u,v)$, on the surface of a cubic NURBS surface is

$$P(u,v) = \mathrm{U}\mathrm{N}_\mathrm{u}\mathrm{H}\mathrm{N}_\mathrm{v}^\mathrm{T}\mathrm{V}^\mathrm{T}$$

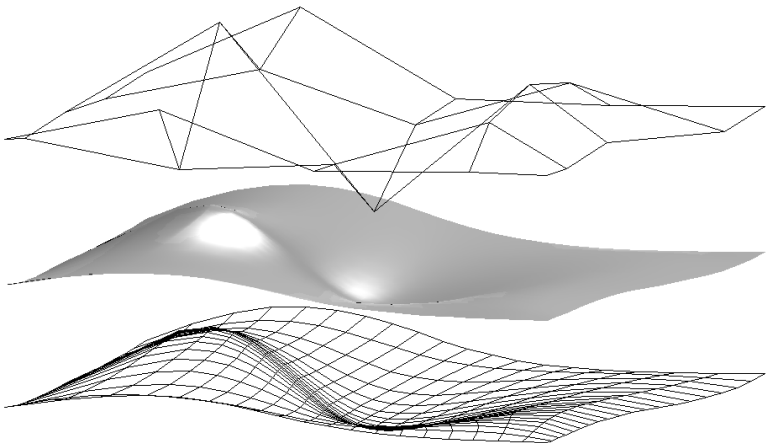

Figure 2: Example of a NURBS surface. The NURBS control points are drawn as a mesh above the surface. Below the surface is the dense mesh of points that would be required to render a similar surface by using polygons.

where

$$
\begin{aligned}
\mathrm{U} &= [1, u, u^2, u^3], \\
\mathrm{V} &= [1, v, v^2, v^3], \\
\mathrm{N_u} &= \text{4 x 4 coefficient matrix for } u, \\
\mathrm{N_v} &= \text{4 x 4 coefficient matrix for } v, \\
\mathrm{H} &= \{\mathrm{C}_{r,c}\} \text{ for } r = i, \ldots, i+3 \text{ and } c = j, \ldots, j+3, \\
&\quad \text{where } u_{i+1} < u \leq u_{i+2} \text{ and } v_{j+1} < v \leq v_{j+2}, \\
\mathrm{C}_{r,c} &= (x_{r,c} w_{r,c}, y_{r,c} w_{r,c}, z_{r,c} w_{r,c}, w_{r,c}), \\
(x_{r,c}, y_{r,c}, z_{r,c}) &= \text{ control vertex at row } r \text{ and column } c \text{ of the control polygon, and} \\
w_{r,c} &= \text{ the weight for the control vertex at row } r \text{ and column } c
\end{aligned}
$$

The coefficient matrices, N_u and N_v, are often calculated using the recursive, knot-insertion algorithm of Boehm [1, 11]. Choi, et al., [2] developed an explicit matrix form for calculating the coefficients of the homogeneous, B-spline blending functions. They show that their procedure has polynomial time complexity in the degree of the surface, while Boehm's method has exponential complexity. Therefore, we use Choi, et al.'s, method, which is now described.

Let the knots for the u parameter be labeled $s_1, s_2, \ldots, s_{m+4}$ and the knots for the v parameter be labeled $t_1, t_2, \ldots, t_{n+4}$, where m and n are the number of control vertices along the u and v directions, respectively, of the control polygon. To find the point $P(u, v)$ on the surface of the NURBS, given parameter values u and v, first determine the pair of (s_i, s_{i+1}) values and (t_j, t_{j+1}) values that include u and v. Given these spans, the coefficient matrices N_u and N_v are given by

$$
\mathrm{N}_u = \begin{pmatrix}
\frac{(s_{i+1}-s_i)^2}{(s_{i+1}-s_{i-1})(s_{i+1}-s_{i-2})} & (1 - n_{11} - n_{13}) & \frac{(s_i-s_{i-1})^2}{(s_{i+1}-s_{i-1})(s_{i+2}-s_{i-1})} & 0 \\
-3n_{11} & 3n_{11} - n_{23} & 3\frac{(s_i-s_{i-1})(s_{i+1}-s_i)}{(s_{i+1}-s_{i-1})(s_{i+2}-s_{i-1})} & 0 \\
3n_{11} & n_{33} - 3n_{11} & 3\frac{(s_{i+1}-s_i)^2}{(s_{i+1}-s_{i-1})(s_{i+2}-s_{i-1})} & 0 \\
-n_{11} & n_{11} - n_{43} - n_{44} & -\frac{n_{33}}{3} - n_{44} - \frac{(s_{i+1}-s_i)^2}{(s_{i+2}-s_i)(s_{i+2}-s_{i-1})} & \frac{(s_{i+1}-s_i)^2}{(s_{i+2}-s_i)(s_{i+3}-s_i)}
\end{pmatrix}
$$

and a similar expression for N_v with t_j substituted for s_i. The terms n_{kl} refer to the element of N_u in the k^{th} row and l^{th} column.

The automatic construction of a smooth surface that approximates a set of points is a difficult problem. Smoothness constraints on each NURBS surface and continuity constraints between surfaces confound the objective of accurately fitting the known surface points. This problem is usually addressed by iterative schemes that incrementally reduce the difference between the surface and the sample points while maintaining the desired continuity conditions. This is a very time consuming process that must be repeated for every patch.

Much of the automatic construction work has dealt with fitting a single smooth surface to the known points. An organic structure can have a complex topology that requires multiple smooth surface patches to be constructed and interconnected. This problem has been dealt with in at least two ways. Guo [9] first uses 3-D α-shapes [7] to find the topology of the surface. The method of α-shapes works well for data sampled from serial contours. The resulting polygon-based surface provides the initial control points for an approximating B-Spline surface, which is then refined by iteratively adjusting the control points to minimize the error between the surface and the sample points. Eck and Hoppe [6, 10] take a different approach. They state that in dealing with organic structures, multiple surface patches must be constructed to handle the complex topologies. Much of their work focuses on refining the initial polygon surface to obtain a mesh of quadrilaterals. The quadrilaterals are used to generate the initial control points for B-Spline patches and these points are iteratively optimized.

Visible Productions has implemented novel methods for refining the polygon surface, both automatically and via user control. The skill of their personnel in sculpting the polygon-based models is attested to by the world-wide recognition of the superiority of their models. Therefore, in this project we will take their polygon-based models as an excellent starting point for our NURBS approximation methods.

4 Approach

4.1 The Initial Approximation of Data by a NURBS Surface

Once a set of cylinders have been identified, each cylinder must be approximated with a NURBS surface. To initialize the process of fitting a NURBS surface to contours of data, we must specify the control points, weights, and knot vectors for the initial NURBS surface. We start by assigning u and v parameter values corresponding to each data point. We base this on the convention of summing arc lengths along the two parameter dimensions.

Let $d_{i,j}$ be the j^{th} data point in contour i. The distance between successive points in the v direction along a contour is

$$\Delta_v d_{i,j} = ||d_{i,j} - d_{i,j-1}||.$$

The total arc length along a contour is found by summing $\Delta_v d_{i,j}$ along the contour. The v parameter value, $v_{i,j}$ for each data point along the contour is defined to be

$$v_{i,j} = \begin{cases} \frac{\sum_{k=1}^{j} \Delta_v d_{j,k}}{\sum_{k=1}^{n_i} \Delta_v d_{i,j}}, & j > 0; \\ 0, & j = 0, \end{cases}$$

where n_i is the number of data points in contour i. This results in v values that range from 0 to 1. The values of $u_{i,j}$ are assigned similarly.

Now the initial control points can be placed. Given that the NURBS surface is defined to have n rows and m columns of control points, we can specify u and v values corresponding to each control point assuming that they will be distributed evenly over the ranges of u and v. Thus, the u and v

values for the control point in row i and column j, are

$$u(c_{i,j}) = \frac{i}{n-1} \quad \text{and} \quad v(c_{i,j}) = \frac{j}{m-1}, \quad \text{for } i = 0, \ldots, n-1 \text{ and } j = 0, \ldots, m-1.$$

Now we find the four data points, d_1, d_2, d_3, and d_4 and their corresponding parameter values, (u_1, v_1), (u_2, v_2), (u_3, v_3), and (u_4, v_4), that form the vertices of the plane containing the parameters of the control point. The coordinates of the control point are then found by linearly interpolating the data points:

$$c_{i,j} = (1 - v_f)\left[u_f d_3 + (1 - u_f)d_1\right] + v_f\left[u_f d_4 + (1 - u_f)d_2\right],$$

where

$$u_f = \frac{u(c_{i,j}) - u_1}{u_3 - u_1} \left(= \frac{u(c_{i,j}) - u_2}{u_4 - u_2} \right)$$

and

$$v_f = \frac{v(c_{i,j}) - v_1}{v_2 - v_1} \left(= \frac{v(c_{i,j}) - v_3}{v_4 - v_3} \right)$$

After the n x m grid of control points is initialized, we add duplicate control points along the $v = 0$ and $v = 1$ ends of the grid to create a smooth join along the v direction, which is along each contour. This specializes the NURBS surface for the types of cylinders we are dealing with in approximating contour data. A total of four control points, two on either side of the join, are added for each row of the control point grid.

All components of the weight matrix, W, are initialized to 1.

The knot vector in the u direction is initialized to the standard non-uniform knot vector for third-degree NURBS curves,

$$[0,\ 0,\ 0,\ 0,\ \frac{i}{n-3} \text{ for } i = 1, \ldots, n-3,\ 1,\ 1,\ 1].$$

The knot vector in the v direction is initialized to the uniform knot vector

$$[\frac{i}{m}, \quad \text{for } i = -4, \ldots, m+3].$$

4.2 NURBS Optimization

We define a NURBS surface to be *optimized* if the mean-squared error, E, between the data points and the corresponding NURBS surface points is a local minimum. E depends on the control points, C, the weights, W, the knot vectors, U and V, and the set of data points D. Let $\hat{u}$ and $\hat{v}$ be the parameter values for the point on the NURBS surface that corresponds to data point d. This notation is also applied to other variables. Then, we can define E to be

$$E(C, W, U, V, D) = \frac{1}{2|D|} \sum_{d \in D} ||d - P(\hat{u}, \hat{v})||^2.$$

Local minima in E can be found by performing a gradient-descent procedure to optimize C and W. (The knot vectors U and V can also be optimized in this way, but we have not yet explored this.) First, we drop the arguments to E for clarity. The gradient of E with respect to each coordinate of the control points, C_i, for $i = 1, \ldots, 3$ for the x, y, and z coordinates, is

$$\nabla_{C_i} E = \frac{-1}{|D|} \sum_{d \in D} \hat{R}(d_i - P_i(\hat{u}, \hat{v})) \cdot \hat{W},$$

where

$$\hat{R} = \frac{\hat{N_u}^T \hat{U}^T \hat{V} \hat{N_V}}{U N_u W N_v^T V^t}$$

and the $\cdot$ operator means component-wise multiplication. Similarly, the gradient of E with respect to the weights, W, is

$$\nabla_W E = \frac{-1}{|D|} \sum_{d \in D} \hat{R} \sum_{i=1}^{3} (d_i - P_i(\hat{u}, \hat{v}))(\hat{C}_i - P_i(\hat{u}, \hat{v}))$$

Gradient descent based on the above gradients will arrive at a local minimum of E. This procedure involves iterations of the following update equations:

$$\begin{aligned} C &\leftarrow C - \alpha_C \nabla_C E \\ W &\leftarrow W - \alpha_W \nabla_W E \end{aligned}$$

4.3 Prediction of Control Point Placement Using a Neural Network

We have invested considerable effort in developing an intelligent approach to the initialization of the control points for a NURBS surface, given the data to be approximated. As the previous figures show, the initial NURBS surface approximates the data well in some places, but not in others. For those areas where the initialization is a poor fit to the data, the gradient-descent procedure requires many iterations to improve the fit. In this section, we describe a novel approach to initialization based on artificial neural networks that estimate the optimal placement of control points. Our current development is in 2-D, but is easily extended to the full 3-D case.

Our approach is to train a neural network to predict where the next control point should be placed, given the previous three control points and the data points that form the part of the data set to be approximated by the three existing control points and the new one to be determined. Examples to train the network were gathered by randomly generating control points and using them to produce data points along the corresponding NURBS surface. 1,000 such 2-D NURBS surfaces were produced, and 4,321 examples were extracted. Each example consists of three control points and 30, 2-D data points. Each example also includes the desired next control point.

A common, feedforward neural network was trained using error backpropagation. The network consisted of 66 inputs, 20 hidden units, and two output units for the two components of the predicted control point. The set of 4,321 examples was partitioned into a training set of 3,456, a validation set of 432, and a test set of 433, roughly an 80%, 10%, and 10% partitioning. After each pass through the training set, or one epoch, the error on the validation set was calculated. After training for 1,000 epochs, the network's weight values at the epoch for which the validation error was the smallest were restored to be used to predict the next control point coordinates for the test set.

5 Results and Evaluation

Figure 3a shows the contours of data points in red, the initial grid of control points in green, and the initial NURBS surface shaded in gray. The data is from hand-traced contours of the right fibula bone of the Visible Human data set. Figure 3b shows the resulting NURBS surface and the final positions of the optimized grid points.

Figure 4 shows a graph of E versus number of iterations. The graph shows that E decreases from an initial value of 24.5 to a final value of 7.5. It continues to decrease only slightly with additional iterations.

a. Initial b. Final

Figure 3: a) Initial NURBS surface (gray) approximating contour data (red) with initial grid of control points (green). b) NURBS surface (gray) after 20 iterations of gradient-descent optimization. The initial control points (red) have been moved to their final positions (green).

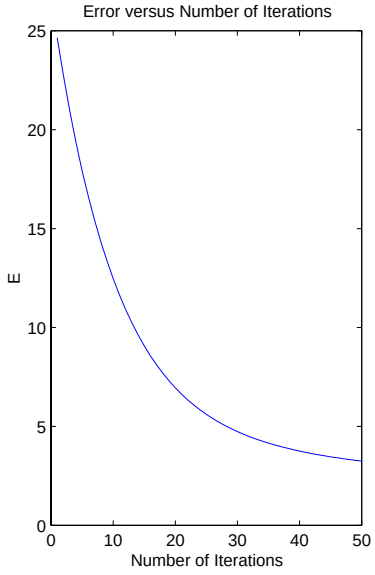

Figure 4: The value of E versus the number of iterations.

To gain an intuitive understanding of the changes made during the optimization process, we have drawn the final NURBS surface with the final control point grid and with the original control point grid. This is shown in Figure 5.

A better view of the NURBS surface is obtained by rendering just the surface with no control points or data points. This is shown in Figure 6.

A key result of the conversion to a NURBS surface is the tremendous reduction in the amount of space required to store the specification of the NURBS compared to the storage of the polygon mesh. For the example shown here, the gzipped file for the polygon mesh requires 955.6 KB, whereas the gzipped file for the NURBS surface requires 10.4 KB, a reduction in size of about 98%.

Now the results of training and using a neural network to predict control point placement are described. Figure 7 shows a graph of the training error and the validation error versus epochs. The best epoch was at Epoch 99. A lower validation error might have been achieved had we trained this network for additional epochs. The final test error is about 0.12, which means that on average the predicted coordinates of the next control point were only off by 0.12. The range of the data was from 0 to 10, so this is approximately a 10% error.

Figure 8 illustrates what this performance level means in terms of how far off the predicted control point placement is from the actual. This figure shows 12 examples from the test set. None of these were used to train the network, yet it does very well at placing the next control point. The green lines show the sequence of control points that were used to generate the data, in red. The fourth control point as predicted by the network is shown as the blue alternative link in the control polygon. For these examples, and the other 400 examples in the test set, the neural network is doing an excellent job at predicting where the next control point should be. If we now use this trained network to specify the initial placement of the control points of a NURBS surface for approximating the data, very little computation will be required of the optimization process because the neural network will have already placed the control points close to their optimal locations.

When actually used to initialize a NURBS, the neural network will be required to generate not just one new control point, but a sequence of them, enough to represent the entire set of data. To test the feasibility of this, we used our trained network to first generate the fourth control point, given the first three, as above. To generate the next, or fifth, control point, the second, third, and

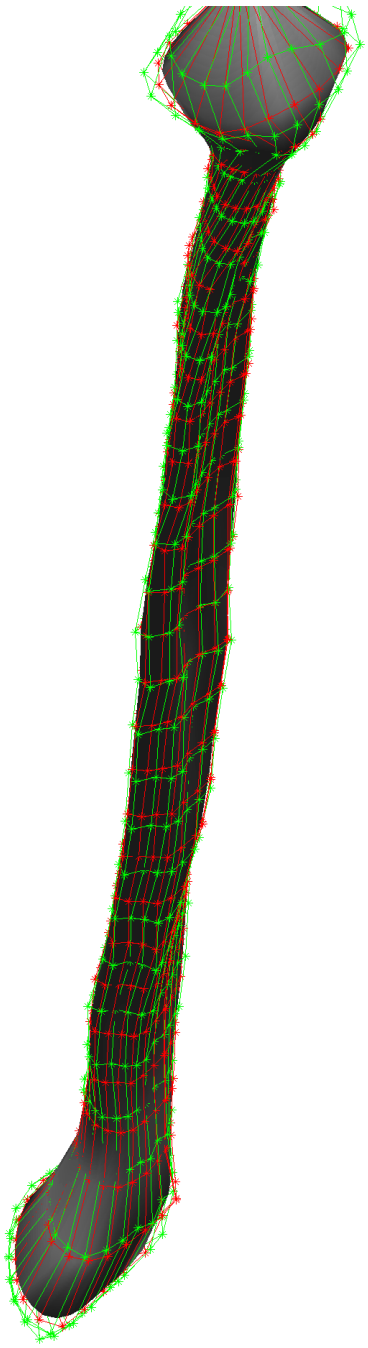

Figure 5: Final NURBS surface with original control points in red and the final control points in green.

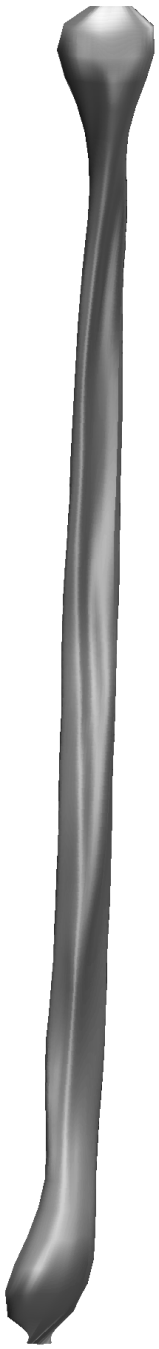

Figure 6: Final NURBS surface approximating the data for the right fibula.

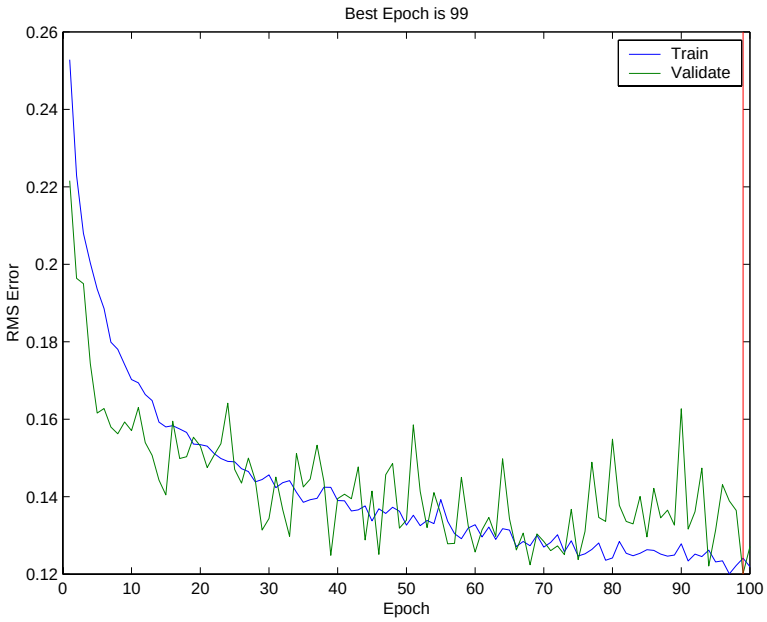

Figure 7: Training RMS error (in blue) and the validation set RMS error (in green) versus the number of epochs. The vertical red line at Epoch 99 shows where the lowest validation error was obtained.

fourth points are input to the network. Note that only the second and third were given a priori; the fourth one was estimated by the network. Once the network has generated the fourth, fifth, and sixth control points, all control points input to the network were predicted by the network. This results in the possibility of errors in the prediction having compounding effects on later control point predictions. However, we find that this has not been the case for the tests we have performed.

Figure 9 shows ten different sets of 2-D points that are to be approximated by a NURBS. The data points, shown in red, were generated from ten different NURBS curves with randomly-generated control points. The first three control points, drawn in blue, were used as the inputs to the neural network for its first prediction. The result is the first control point drawn in black. This prediction and the previous two control points are used to predict the next control point. As shown in the figure, the sequence of predicted control points, in black, are close to the actual control points, drawn in green. These predicted control points are much closer to their optimal placement than would control points generated by interpolating the control points, the procedure described earlier in this report.

The success of predicting control point placement in 2-D is very encouraging. This can easily extended to the 3-D case to place the initial locations of control points on a grid. This will be a major effort in the continuation of this project.

5.1 Writing NURBS Surface to IGES File and Manipulation with Standard CAD Tools

One goal of the conversion of Visible Production's polygonal models to NURBS models is to better interface with customers. Aside from straightforward visualization purposes, some clients want the ability to manipulate the models in a CAD package. There are many CAD package possibilities, such as IronCAD, AutoCAD, Solid Works, CADKEY, and Solid Designer. Instead of dealing with the native formats for each of these packages, we decided to use the IGES format.

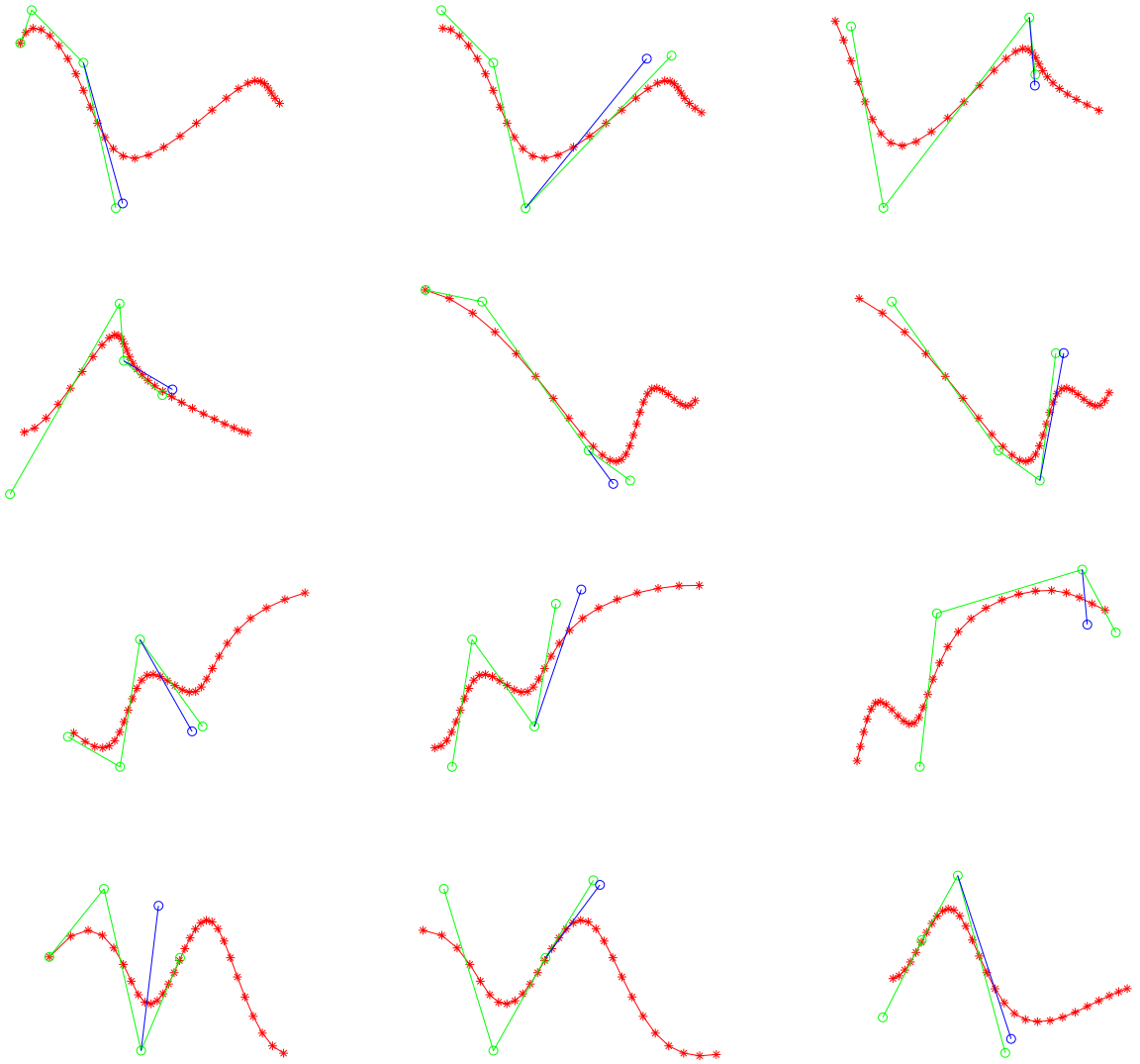

Figure 8: 12 examples from the test set. The control points used to generate the data are shown in green, and the data in red. Given the first three control points and the data shown, the neural network predicts the location of the fourth control point to be in the location shown in blue. Numbering these examples from the upper left and across, the first four examples are from the data curve, the next two are from a different one, the next three are from one curve, and the final three are from one curve.

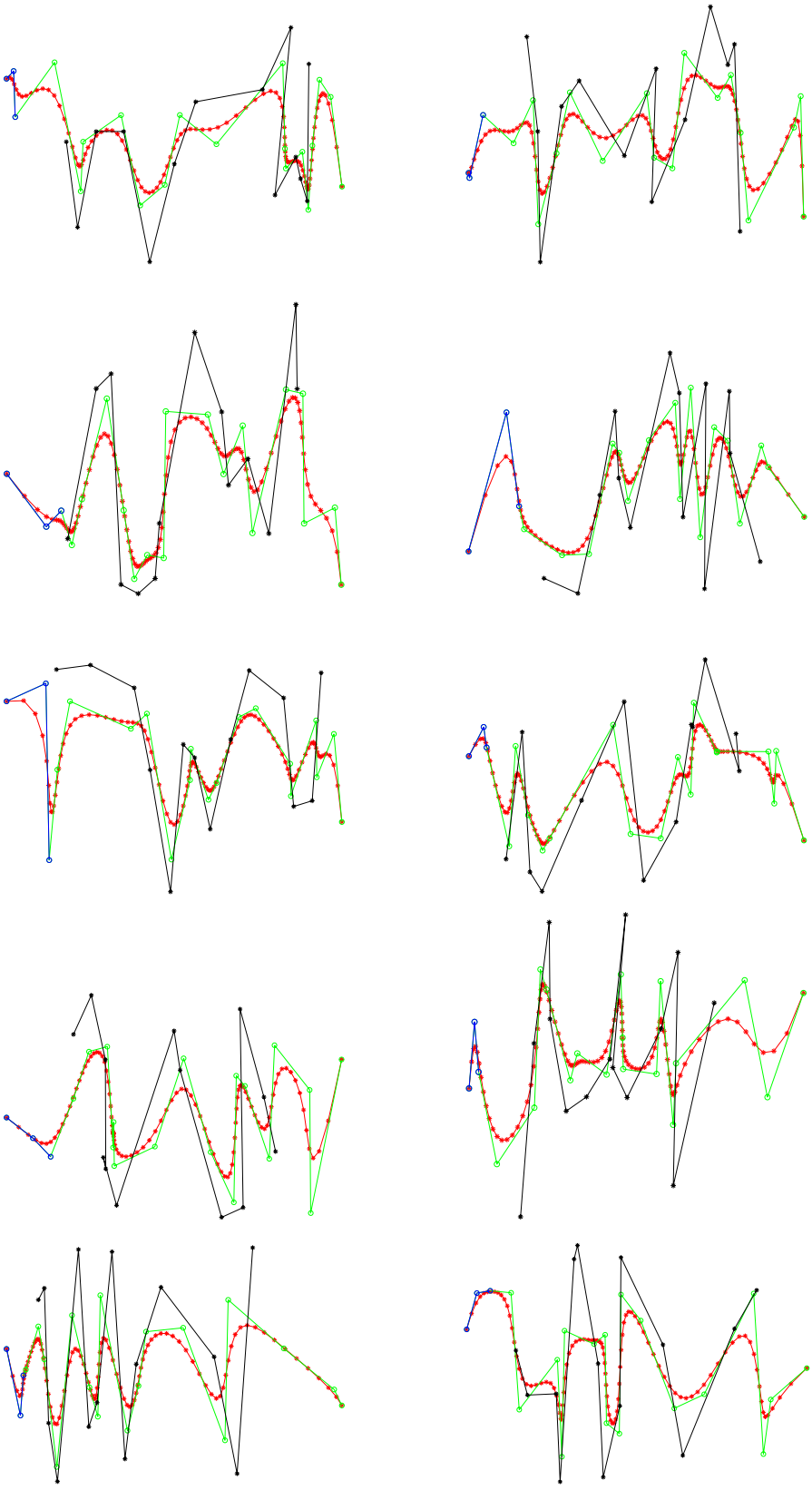

Figure 9: Tests

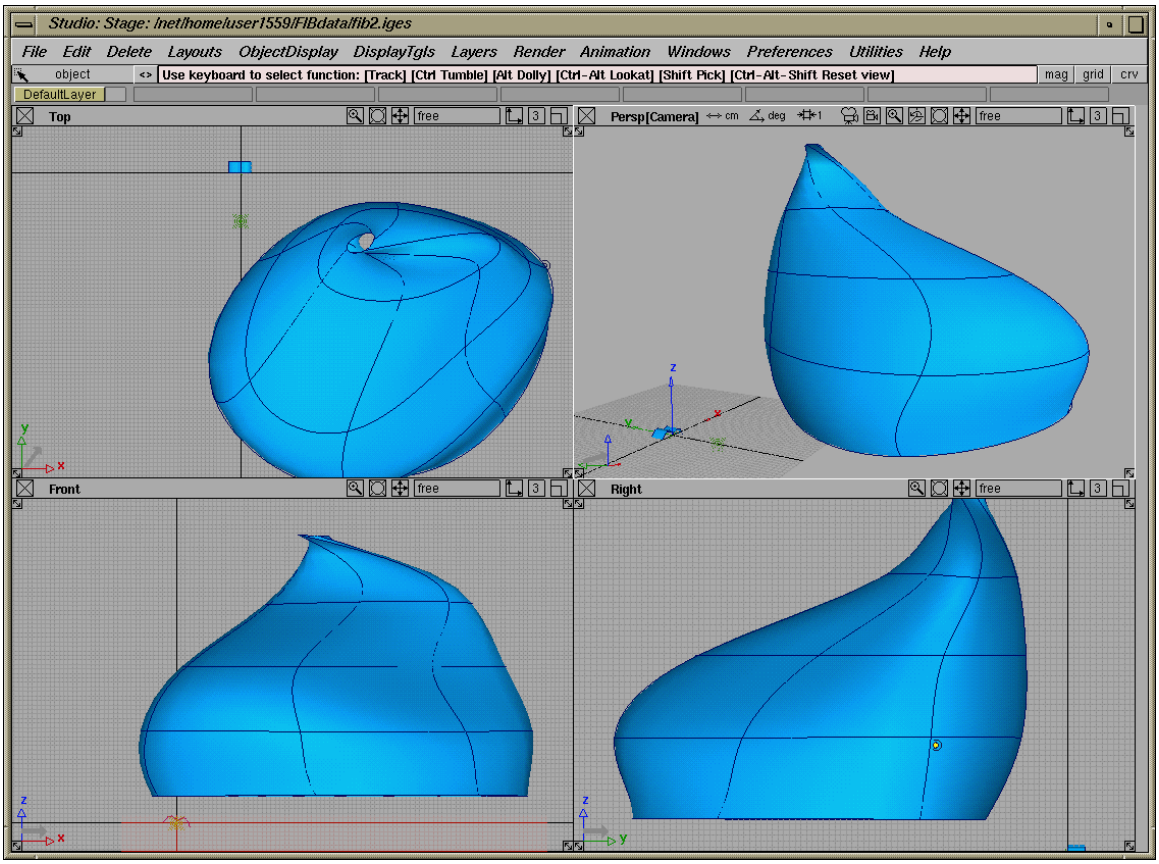

Figure 10: Alias Studio after loading a section of one of our NURBS surfaces that was stored in IGES format.

IGES (Initial Graphics Exchange Specification) is a standard format which most CAD packages can import and is a widespread standard for CAD data exchange [12]. Another key reason for using the IGES standard is that it is an open standard, not beholden to the proprietary interests of any one product organization. The standard is coordinated and published by NIST and is a recognized ANSI standard. (The current standard is up to version 5.2, though the NURBS specification section we needed has not changed substantially from the version 4.0 standard cited).

Figure 10 shows one end of the NURBS surface that we generated to approximate the fibula data after the NURBS file is loaded into by Alias Studio. At this point, a user may directly manipulate the control points to modify the surface to meet their needs. This obviates the cumbersome polygon selection processes required to manipulate polygon meshes.

6 Conclusion

The feasibility of using neural networks to initialize the process of optimizing the fit of NURBS surfaces to 3-D data points was demonstrated by the experiments reported here. The current results, though, are limited to 2-D data points. The next step of proving the feasibility of the approach to a full 3-D data set remains to be taken.

References

[1] W. Boehm. Inserting new knots into b-spline curves. *Computer-Aided Design*, 12(4):199–201, 1980.

[2] B. K. Choi, W. S. Yoo, and C. S. Lee. Matrix representation for nurb curves and surfaces. *Computer-Aided Design*, 22(4):235–240, May 1990.

[3] S. Crawford-Hines and C. W. Anderson. Interactive region bounding with neural nets. In *NNACIP'94—International Workshop on Neural Nets Applied to Control & Image Processing*, pages 58–61. Mexican Association of Automatic Control (AMCA) and IEEE, November 1994.

[4] S. Crawford-Hines and C. W. Anderson. Neural nets in boundary tracing tasks. In J. Principe, L. Giles, N. Morgan, and E. Wilson, editors, *Neural Networks for Signal Processing VIII, Proceedings of the 1997 IEEE Workshop*, pages 207–215, 1997.

[5] S. Crawford-Hines and C. W. Anderson. Machine learned contours to assist boundary tracing. In *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation*, Tucson, AZ, 1998.

[6] M. Eck and H. Hoppe. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 325–334. ACM, 1996.

[7] H. Edelsbrunner and E. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43–72, 1994.

[8] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes, and R. L. Phillips. *Introduction to Computer Graphics*. Addison-Wesley Publishing Company, 1994.

[9] B. Guo. Surface reconstruction: From points to splines. *Computer-Aided Design*, 29(4):269–277, 1997.

[10] H. Hoppe. *Surface Reconstruction from Unorganized Points*. PhD thesis, Department of Computer Science and Engineering, University of Washington, Seattle, WA, 1996.

[11] L. Piegl and W. Tiller. *The NURBS Book*. Springer-Verlag, New York, 1997.

[12] B. Smith, G. Rinaudot, K. Reed, and T. Wright. Initial graphics exchange specification (IGES) version 4.0. US Dept of Commerce, National Bureau of Standards (now NIST) NBSIR 88-3813, June 1988.