1997

Feature-Based Models for Three-Dimensional Data Fitting.

Gregory Todd Dobson

Louisiana State University and Agricultural & Mechanical College

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_disstheses

Recommended Citation

https://digitalcommons.lsu.edu/gradschool_disstheses/6562

This Dissertation is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Historical Dissertations and Theses by an authorized administrator of LSU Digital Commons. For more information, please contact gradetd@lsu.edu.
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6” x 9” black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
FEATURE-BASED MODELS FOR THREE-DIMENSIONAL DATA FITTING

A Dissertation

Submitted to the Graduate Faculty of the Louisiana State University and Agricultural and Mechanical College in partial fulfillment of the requirements for the degree of Doctor of Philosophy

in

The Department of Mechanical Engineering

by

Gregory Todd Dobson
B.S., Louisiana State University, 1990
December 1997
Acknowledgements

First, I would like to thank my parents, Judy and Earl Dobson, for their unwavering love and support throughout my life. Their sacrifices and encouragement have allowed me to achieve all of the goals I have set for myself. I am the person I am today because of them and, for this, I will be eternally grateful. Additional thanks and love are extended to my brothers, sister and many family members whose confidence and support over the years are cherished immeasurably.

I would also like to express my esteem for the numerous professors who have at one time or another participated on my graduate committee. These are Doctors Sumanta Acharya, Michael Murphy, Daniel Yannitell, Neal Stoltzfus, Warren Liao, David Thompson, Vic Cundy, Dirk Smith, Robert Dorroh, and Professor A.J. McPhate. Of these, I would especially like to thank Doctors Acharya, Yannitell and Murphy for the many helpful suggestions they have made in editing this manuscript. Perhaps the most influential person in my academic life has been my major professor, Doctor Warren Waggenspack, Jr. His academic and professional guidance has vastly affected this research and my budding career.

A number of companies have supported this research and need to be recognized. Without the use of the computer equipment provided by Munro Shoe Company and Applied Geometry, LLC, this project would not have been completed in such a timely fashion. Also, essential data sets were provided by Cyberware Scanning Systems, Brightwood Shoes, Red Wing Shoe Company and Laser Design, Inc. Their generosity contributed tremendously to the
completion of this work. The research was also supported in part by a fellowship from the L.S.U. Board of Regents.

I must also offer a debt of gratitude to the many individuals who have been a part of the Interactive Modeling Research Laboratory over these many years of graduate work. They have provided endless hours of advice, feedback and much-needed distraction during my tenure in the lab. These individuals include: Sandeep Dani, Tarek Zohdi, Siddarameshwar Bagali, Olalekan Odesanya, David Thompson, Yohannes Desta and John Fuller. Special thanks go to Gary Templet, Anirban Bardalaye, Seshu Simhadri and Robert Benton whose help on the day of my defense is greatly appreciated. I must also thank system managers Aleric Haag and Tyrone Schultz for keeping the system alive and running during the development of this project.

Of the members in the lab, I must sincerely thank Henry Lamousin for his friendship throughout my graduate student life. His no-nonsense commentary and stoic realism (bordering on cynicism) has keep me focused on the job at hand, no matter what that job may be. I owe much of my success and my sanity to his wisdom and support. Thank you!

Finally, I must convey my deepest thanks and appreciation to my partner in life, Lisa Carroll. Her patience and caring throughout this ordeal called graduate school can only be compared to that of Job. For the rest of my life, I can only hope to return a fraction of the love and kindness I have received over the first two years of our marriage. I will always love you.
Table of Contents

Acknowledgements........................................................................................................ii
List of Tables..................................................................................................................vi
List of Figures...............................................................................................................vii
List of Symbols...........................................................................................................xi
Abstract.......................................................................................................................xiv

Chapter 1 Introduction..............................................................................................1

Chapter 2 Literature Review ..................................................................................4
  2.1 Surfacing Descriptions...................................................................................5
  2.1.1 Triangulated Surfaces ...........................................................................6
  2.1.2 Contour Representations .......................................................................8
  2.1.3 Volumes .................................................................................................10
  2.1.4 Algebraic Polynomials ........................................................................10
  2.1.5 B-Splines ...............................................................................................12
  2.2 Approximation / Interpolation Solutions ..............................................15
  2.2.1 Explicit Solutions..................................................................................16
  2.2.2 Parameterization of Data Points..........................................................18
  2.3 Adjustment Techniques ............................................................................19
  2.3.1 Iterative Addition/Subtraction ............................................................19
  2.3.2 Adaptive Subdivision ..........................................................................20
  2.3.3 Closest Point Calculations ..................................................................22
  2.3.4 Iterative Adjustment Techniques .........................................................23
  2.4 Preliminary Modeling Systems .................................................................25
  2.5 Registration .................................................................................................27
  2.5.1 Landmarks............................................................................................27
  2.5.2 Moments ...............................................................................................28
  2.5.3 Optimization .........................................................................................29
  2.6 Nonlinear Systems (Optimization) ............................................................31
  2.6.1 Cost Function Norms ..........................................................................31
  2.6.2 Conjugate Gradient Methods .............................................................33
  2.6.3 Newton-Type Methods .......................................................................34
  2.6.4 Mixed Systems .....................................................................................37
  2.7 Summary ......................................................................................................38
### Chapter 3 Feature-Based Fitting Methodology

3.1 Model Construction .................................................................................. 40
3.2 Model Registration
   3.2.1 Initial Alignment .................................................................................. 43
   3.2.2 Closest Point Calculations .................................................................. 44
   3.2.3 Optimized Registration ...................................................................... 45
3.3 Feature-Based Deformation ...................................................................... 46
3.4 Vertex Optimization ................................................................................. 49
3.5 Hierarchical Refinement .......................................................................... 50
3.6 Sample Applications and Results
   3.6.1 Pilot Study (2-D Facial Profiles) ......................................................... 53
   3.6.2 Three-Dimensional Surface Fitting .................................................... 64
3.7 Summary .................................................................................................. 77

### Chapter 4 Algorithm Testing

4.1 Registration ............................................................................................. 79
4.2 Feature Deformation ............................................................................... 90
4.3 Tolerance Fitting .................................................................................... 92
4.4 Repeatability and Accuracy Tests ........................................................... 94
4.5 Summary .................................................................................................. 99

### Chapter 5 Summary and Conclusions

5.1 Advantages ............................................................................................. 101
5.2 Limitations and Future Work
   5.2.1 Alignment .......................................................................................... 103
   5.2.2 Feature Deformation ......................................................................... 103
   5.2.3 Vertex Optimization ......................................................................... 104
   5.2.4 Miscellaneous Additions .................................................................... 105
5.3 Potential Applications ............................................................................. 106

### Bibliography

109

### Appendix: Additional Data Sets

128

### Vita

135
List of Tables

Table 3.1 Maximum distances (inches) from the model to data and CPU times (sec) on a VAX 7600 for each stage of the feature-based modeling process..............................................................56

Table 3.2 Feature point distances for the sample profiles in Figures 3.10-3.14. All measures are in inches.................................63

Table 3.3 Maximum errors (inches), number of iterations and run times (sec) on a Silicon Graphics O2 machine (180 MHz) for the four feet shown in Figure 3.24, and Figures 3.27-3.29 (Feet 1-4, respectively)............................................75

Table 3.4 Extracted measures for the data in Figures 3.24 and 3.27-3.29, respectively. All values are in inches...........................76

Table 4.1 Initial position and rotation angles (degrees) for the four test data sets. The associated differences and relative, percentage errors from the optimized values are also given. The position percentages are scaled relative to an 11.2" foot length and the angular percentages to a full 360 degree rotation.......................................................86

Table 4.2 Optimized alignment parameters for six versions of the same foot scan. Alignment rotation angles are in degrees. Position percentages are relative to a foot length of 11.7 inches and the angle percentages are relative to a full 360 degree rotation......................................................89

Table 4.3 Run times (sec.), squared errors, and maximum error (inches) for the six data sets in Table 4.2 using the four different variations of the vertex optimization routine.............................................94

Table 4.4 Results of repeatability test for the data sets given the Appendix. All measures are in inches.................................95

Table 4.5 Supplied and calculated measures for Feet 3 and 4. All measures are in inches. The associated differences and percentage errors are also provided.................................................97

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
List of Figures

Figure 2.1  Point cloud representing a human foot.................................................6
Figure 2.2  Triangulated surface constructed from the data set shown in Figure 2.1 ..................................................................................7
Figure 2.3  Slice data set connected linearly between contours. .......................8
Figure 2.4  Bi-Cubic, Non-uniform Rational B-spline Surface with 3 patches ..........................................................................................14
Figure 2.5  Example of a free-form deformation on a cylinder..........................24
Figure 3.1  Facial profile with 8 feature points marked. .................................42
Figure 3.2  Picture of the nose feature with parametric range and associated closest point error vectors ..................................................47
Figure 3.3  Picture of nose with average error vector denoting required deformation .............................................................................48
Figure 3.4  B-spline curve span (bolded) to be subdivided using hierarchical refinement...........................................................................51
Figure 3.5  Hierarchical refinement of central span. New vertices are shown as large open circles.................................................................52
Figure 3.6  Deformed hierarchical curve span. The original curve is shown as a dashed line. .................................................................52
Figure 3.7  Facial profile model comprised of 16 control vertices. The vertices are offset horizontally for clarity ..................................54
Figure 3.8  Alignment and feature fitting stages prior to tolerance fit for Profile 1: (a) Alignment and initial sizing; (b) Feature point deformation. ..................................................56
Figure 3.9  Alignment and feature fitting stages prior to tolerance fit for Profile 2: (a) Alignment and initial sizing; (b) Feature point deformation. ..................................................57
Figure 3.10 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 1. ......................................58

Figure 3.11 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 2. ..............................59

Figure 3.12 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 3. ..............................60

Figure 3.13 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 4. ..............................61

Figure 3.14 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 5. ..............................62

Figure 3.15 Iso-parametric slices from the data set given in Figure 2.1 ...................................................................................................65

Figure 3.16 Surface constructed using a least squares fit to the slices given in Figure 3.15 ......................................................................65

Figure 3.17 Defined feature points with associated parametric ranges ..................................................................................................66

Figure 3.18 Point-to-point measures and ball and waist girths. The model is displayed as a grid of patches for ease of viewing. ........................................................................................................66

Figure 3.19 Projected measures for a foot. The model is displayed as a grid of patches for ease of viewing. ........................................67

Figure 3.20 Foot model and slice data set positioned as defined. .............................69

Figure 3.21 Initial alignment of the foot model to the data using center of mass and moment calculations only..............................69

Figure 3.22 Optimized alignment for the model and data set shown in Figure 3.21 ........................................................................70

Figure 3.23 Feature point deformation for the model and data set shown in Figure 3.22 ......................................................................71

viii

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 3.24 Final tolerance fit for the model and data set shown in Figure 3.23 .................................................................72

Figure 3.25 Point-to-point distances and girth (circumference) measures for the final fit shown in Figure 3.24.................................72

Figure 3.26 Projected measures for the fit shown in Figure 3.24 ....................................................................................73

Figure 3.27 Final tolerance fit for Foot 2. ...............................................................................................................74

Figure 3.28 Final tolerance fit for Foot 3. ...............................................................................................................74

Figure 3.29 Final tolerance fit for Foot 4. ...............................................................................................................75

Figure 4.1 Data set consisting of 4088 data points. The data set is used in the initial alignment testing of Section 4.1 ...............80

Figure 4.2 Data set consisting of 1,095 data points. The data set is used in the initial alignment testing of Section 4.1 ..............81

Figure 4.3 Initial alignment of the data set consisting of 1095 points, taken as a point cloud. ..................................................82

Figure 4.4 Initial alignment of the data set consisting of 4088 points, taken as a point cloud. ..................................................83

Figure 4.5 Initial alignment of the data set consisting of 1095 points, taken as a series of slices ...............................................84

Figure 4.6 Initial alignment of the data set consisting of 4088 points, taken as a series of slices ...............................................85

Figure 4.7 Optimized alignment of the second data set in Table 4.2 (Figure A.2 in the Appendix). The surface is displayed as patch lines for clarity .................................................................90

Figure 4.8 Feature point deformation of the model in Figure 4.1. The image shows a localized bulge at the inner arch and a fold at the top of the heel .............................................................91

Figure 4.9 Pinching of the model surface due to an initial misfit at the inside of the toe box .....................................................92

ix

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 4.10  Girth and point-to-point measures for Foot 3.................................97

Figure 4.11  Projected measures for Foot 3 ......................................................98

Figure 4.12  Girth and point-to-point measures for Foot 4.................................98

Figure 4.13  Projected measures for Foot 4 ......................................................99

Figure A.1  Version number 1 of the data set used in the feature
deformation, tolerance fitting and repeatability
testing (3,109 data points) .................................................................129

Figure A.2  Version number 2 of the data set used in the feature
deformation, tolerance fitting and repeatability
testing (4,713 data points) .................................................................130

Figure A.3  Version number 3 of the data set used in the feature
deformation, tolerance fitting and repeatability
testing (17,091 data points) .................................................................131

Figure A.4  Version number 4 of the data set used in the feature
deformation, tolerance fitting and repeatability
testing (2,353 data points) .................................................................132

Figure A.5  Version number 5 of the data set used in the feature
deformation, tolerance fitting and repeatability
testing (9,965 data points) .................................................................133

Figure A.6  Version number 6 of the data set used in the feature
deformation, tolerance fitting and repeatability
testing (5,179 data points) .................................................................134

Republished with permission of the copyright owner. Further reproduction prohibited without permission.
List of Symbols

\( \sigma \) penalty scale factor for constraining vertex weights

\([B]\) matrix of evaluated basis functions for use in linear system solution of control vertex positions

\( B_{i,r} \) Basis function \( i \) of order \( r \)

\( C(x) \) cost function for minimization, \( x \) represents the variable list

\([CP]\) matrix of closest points on the model surface to each data point, each point is placed in a row with the \( x,y,z \) components in the columns

\([D]\) matrix of data points, each point is placed in a row with the \( x,y,z \) components in the columns

\( \delta_u, \delta_v \) span markers, denotes current knot span for evaluation in either the \( u \) or \( v \) parametric direction

\( \Delta F \) matrix of feature point offsets within the feature-point deformation routine

\( f(x) \) general, scalar function of variable vector \( x \)

\( G \) second gradient matrix of cost function

\( g_k \) gradient vector at iteration \( k \) in a conjugate gradient optimization

\([H]\) second derivative, Hessian matrix for use in Newton optimization, \( J^T J + G \)

\( J \) Jacobian matrix, matrix of first partial derivatives of cost function

\( k \) step for iteration

\( l_1 \) least absolute residual norm, Equation [2.7]

\( l_2 \) least squares norm, Equation [2.9]

\( l_\infty \) infinity norm, also known as the min-max norm, Equation [2.8]

\( m \) number of equations in a linear system solution or an optimization
$N_{\text{data}}$  number of points within a given data set

$N_{\text{vertices}}$  total number of vertices defining a B-spline curve/surface

$n$  number of unknowns in either a linear system solution or an optimization routine

$P_i$  $i^{th}$ data point, represented as a column vector

$p$  exponent used in the data parameterization equation [2.5]

$R$  residual vector; array of error differences for minimization

$r_i$  $i^{th}$ component of the residual vector, $R$

$r$  order of basis function, degree of basis function plus 1

$S(u,v)$  parametric B-spline surface point

$S_{cx}, S_{cy}, S_{cz}$  scale factors along the x, y, and z axes, respectively

$s_k$  search direction in a conjugate gradient optimization or the step vector in a Newton optimization at iteration $k$

$[T]$  transformation matrix to align a model to a data set

$[T_r]$  translation matrix, used in the registration of a model to a data set

$U$  normalized, x principal axis for either the data set or model

$u$  parameter value for a B-spline curve; first parameter value (direction) for a B-spline surface

$V$  normalized, y principal axis for either the data set or model

$\Delta V$  matrix of vertex offsets in the feature-point deformation routine; calculated using equation [3.5]

$V_{ij}$  control vertex within the matrix of vertices defining a B-spline surface; $i$ denotes the position in the $u$ direction and $j$ denotes the position in the $v$ direction

$v$  second parameter value (direction) for a B-spline surface

$W$  normalized, z principal axis for either the data set or model
\( w_{i,j} \) weight associated with control vertex \( V_{i,j} \)

\( \mathbf{x} \) vector of unknown variables; used to represent the unknowns in either a linear system or optimization solution

\( \mathbf{x}_k \) solution vector of the unknowns at iteration \( k \) of either a conjugate gradient or Newton optimization

\( \mathbf{x}^* \) value of the unknowns at which the cost function reaches its minimum for a conjugate gradient or Newton optimization
Abstract

There are numerous techniques available for fitting a surface to any supplied data set. The feature-based modeling technique takes advantage of the known, geometric shape of the data by deforming a model having this generic shape to approximate the data. The model is constructed as a rational B-spline surface with characteristic features superimposed on its definition. The first step in the fitting process is to align the model with a data set using the center of mass, principal axes and/or landmarks. Using this initial orientation, the position, rotation and scale parameters are optimized using a Newton-type optimization of a least squares cost function. Once aligned, features embedded within the model, corresponding to pertinent characteristics of the shape, are used to improve the fit of the model to the data. Finally, the control vertex weights and positions of the rational B-spline model are optimized to approximate the data to within a specified tolerance. Since the characteristic features are defined within the model at creation, important measures are easily extracted from a data set, once fit. The feature-based modeling approach is demonstrated in two-dimensions by the fitting of five facial, silhouette profiles and in three-dimensions by the fitting of eleven human foot scans. The algorithm is tested for sensitivity to data distribution and structure and the extracted measures are tested for repeatability and accuracy. Limitations within the current implementation, future work and potential applications are also provided.
Chapter 1 Introduction

The introduction of computers into industry has revolutionized methods for creating and redesigning products. Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) allow engineers to view concepts in three dimensions, make adjustments to their designs and store the plans within reasonably accessible databases. Unfortunately, computerized models do not exist for designs made prior to the introduction of computers into industry. In addition, prototype designs may be manually modified to mesh with existing parts or to create a more functional or aesthetically pleasing form, leaving them different from the current computer model [SARK91a]. Therefore, the new design must be captured and integrated into the existing computerized plan or model.

Reverse engineering refers to the process of creating engineering models, such as two-dimensional (2-D) orthographic drawings or three-dimensional (3-D) surfaces, for existing parts by acquiring the object’s shape, creating the model representation and extracting needed information [KWOK91, WOHL95]. The first step in the process is the acquisition of the shape of the part. A wide variety of methods exist for obtaining points on the surface of an object. These range from manually-controlled coordinate measuring machines (CMM) to laser scanning devices capable of collecting 15,000 surface points per second [PUTT94]. Initially, these devices where used to test the quality of manufacturing processes by comparing actual dimensions to those of the component as designed [RAAB94].

Once surface points have been collected, a three-dimensional surface representation is typically fit to the data. There are numerous, generic
techniques available for fitting surfaces to an arbitrary data set. However, current methods do not take full advantage of the fact that in many applications the basic shape is known. Many of these techniques "blindly" fit patches to the data, resulting in distinctly different surface representations when fitting numerous sets of the same shape. For example, scanning a single object in two different orientations can produce two very different models. Even reordering the points within a single data set may affect the final structure of the surface representation. Extracting information from such diverse models becomes difficult, especially when the needed information must be collected by user-intensive, interactive techniques.

The feature-based modeling method presented in this thesis takes advantage of the fact that, often times, the data represent a known shape. Because the data may be an arbitrary collection of points, a generic model having this known shape is a logical starting point for a consistent curve or surface fitting process. Pertinent features defining the shape are incorporated into the basic structure of the model. Using these features and the concepts described in this thesis, the model is then deformed to fit the unique characteristics of each data set to within a specified tolerance.

The need to run time consuming feature extraction algorithms on the discrete data is eliminated since the fitting process maintains the basic structure and feature values of the model are inherent to its definition. For physicians, this modeling approach can potentially provide an efficient and relatively straightforward method for tracking progressive changes in anatomical structures as well as form a basis for comparisons between different types of medical imaging techniques [ALLA93, BAJC89, FABE88, HOU92, HOUS92, REIB97, ROBB89, TOPP90, TORR92]. Additionally, a structured model
provides consistent space requirements for the data, independent of the actual
scan resolution, and presents some unique opportunities for data compaction.

Components of the feature-based fitting algorithm to be presented have
been discussed to varying degrees in the literature, including: surface
representations, fitting techniques, methods for deforming existing models,
alignment of these models to data sets, and optimization theory. A review of
these topics is presented in the following chapter along with the benefits and/or
limitations of each method for the defined task. Those already familiar with
these topics may skip to Chapter 3 which brings all of these components
together, with reasons for their inclusion, and discusses the innovations needed
to complete the feature-based data fitting process. Chapter 3 also discusses two
distinct applications of the concepts: the first is a pilot study which fits a two-
dimensional facial profile to silhouettes, and the second uses a foot model to fit
3-D scanned data. With the functionality of the approach established, Chapter 4
examines the sensitivity and robustness of the algorithm components to various
data specifications. Lastly, in Chapter 5, the feature-based fitting approach is
summarized and potential applications and future enhancements are suggested.
Chapter 2  Literature Review

There are volumes of information in the literature concerning three-dimensional data fitting. The methods can differ in the surface representation selected, the method for constructing a surface, and/or the techniques for deforming previously created surfaces. The first section of this chapter presents the various surface descriptions implemented by the referenced surface fitting algorithms. This section ends with a discussion of the B-spline surfaces which are used to define the models in the new feature-based fitting process.

The remainder of the chapter will focus on those data fitting techniques that apply directly to B-spline surface fitting. Section 2.2 presents the techniques for constructing a B-spline surface to either interpolate or approximate a given data set. The third section summarizes a myriad of techniques for improving the fit of a constructed B-spline surface to a data set.

The surface construction and improvement methods of Sections 2.2 and 2.3 produce quality fits to a data set. However these techniques do not take advantage of the underlying shape of the data when known a priori. Therefore, Section 2.4 presents the initial attempts of researchers to fit an existing surface model, having this shape, to a supplied data set. The shape-model may be constructed using any one of the techniques given in Section 2.2. Because the model and data set may have very different initial orientations, the first step in the fitting process is aligning the model to a data set, the topic of Section 2.5.

The next step in the process is the deformation of the model to fit the data to a supplied tolerance. Any one of the techniques mentioned in Section 2.3 may be implemented; however, the feature-based fitting algorithm is based
on optimization. Therefore, Section 2.6 discusses a number of optimization routines and the norms used in creating an appropriate cost function.

The chapter ends with a discussion of the techniques utilized in the feature-based fitting algorithm with reasons for their incorporation. A reader familiar with any one of the topics covered may skip that section without loss of continuity.

2.1 Surfacing Descriptions

As mentioned previously, there are numerous data acquisition systems available for collecting points from the surface of an object. Coordinate measuring machines [HSIE93, KWOK91, PUTT94, RAAB94], laser scanners [BAO92, MILR95, REIB97, WILK88] and structured light/CCD systems [BHAT94, DUNC86, LEE96, MAMM96, YOUN94] are commonly used to digitize external surfaces. Internal structures are captured with voxel-based models using ultrasound [BELO92, GIES91, MOCH90, TURN93, WELL88, WELL93], computed tomography (CT) or magnetic resonance imaging (MRI) equipment [ALLA93, BHAT94, PELI89]. Once collected, these data sets are usually very large and dense and do not lend themselves to easy viewing.

In order to improve visualization and provide a compact representation, surfaces are often fit to the data [ATES93, BAJA94, LIM95, ODES93, ROBB89]. Furthermore, the surface model forms a basis for feature extraction [ALLA93, FABE88, ROBB89], data comparisons [HAUS90, HUAN96, TOEN90] and computational analysis [VANO78] which are not readily possible from the data directly. The remainder of this section discusses the various kinds of surface representations and their associated merits. The following sections present the algorithms available for fitting these surfaces to an arbitrary collection of data and deforming previously defined surfaces.
2.1.1 Triangulated Surfaces

In many applications, a data set is compiled by making multiple scans of an object in different orientations. This insures that the entire surface is captured; however, the resulting data set has no general point-to-point associativity. It is simply a cloud of three-dimensional coordinates collectively representing the surface, as shown in Figure 2.1. To provide order to the data, algorithms incorporating Voronoi diagrams and Delauney triangulation can be used to establish the point topology defining the surface [AGIS91, KOLB95, LIM95]. Consequently, triangular facets are easily constructed between sets of three neighboring points on the surface. One triangulation of the point cloud given in Figure 2.1 is shown in Figure 2.2.

Figure 2.1 Point cloud representing a human foot.
The WRAPPER algorithm, presented by Gueziec and Dean, is one such package that creates a tessellated surface from cloud points [GUEZ94]. The resulting surface is a simple, geometric representation and calculations based on this surface, such as area and volume, are readily performed [ALLA93, VANO78]. Since this method is an interpolant, it accurately represents the data; however, the additional topology increases the size of the model significantly. Moreover, the quality of the surface representation depends directly on how accurately the data set captures the shape of the object. Non-uniform densities within the point cloud will result in widely varying facet sizes which may adversely affect visualization and subsequent analyses.

To combat these problems, the WRAPPER software systematically combines and reorients the facets to keep the aspect ratios and sizes of the
triangles similar while fitting the data to within a specified tolerance [GUEZ94]. Other methods deform and subdivide a simple triangulated surface (a sphere for example) to approximate the data while keeping the size of the approximation in control [CHEN94, FOLE90, PRAT93]. Although these techniques adequately fit the data, the resulting surface remains a random collection of facets, providing no consistent model.

2.1.2 Contour Representations

In the medical field, data is often collected and viewed as a set of parallel slices. Contours are segmented from CT, MRI, and ultrasound scans for quantifying lengths, areas and volumes. Any surfacing routine should take advantage of the structure provided. Simply stacking the piecewise-linear contours does not create a surface. The space between the slices are often interpolated linearly by connecting corresponding points of neighboring contours to produce a surface, as seen in Figure 2.3 [SIU96, WOOD87]. Often, this requires that every slice contain the same number of data points.

Figure 2.3 Slice data set connected linearly between contours.
Triangular facet interpolation between the slices allows varying numbers of points per slice [ALLA93, BAJA94, CHEN90, HAUS90, HESS92, MOSH94]. A method presented by Levesque even handles bifurcating regions within a data set [LEVE89]. Since these techniques interpolate the data, the quality of the representation depends directly on the density of the data set and the spacing between slices. Widely varying densities and spacing produce non-uniform facet sizes which may adversely affect numerical analyses. Also, the added topology to produce the surface may double the size of the original data set.

To overcome some of these liabilities, researchers have proposed fitting two-dimensional curves to each contour as a basis for generating equally spaced and numbered data collections [LIN89, ODES93, PARK96]. The posed slice interpolation methods may then be applied, resulting in uniformly sized facets. This approach does allow for data reduction by minimizing the number of points per slice; however, the spacing between the slices is not addressed. Odesanya, et al. suggest blending the curve representations across the slices [ODES93]. This results in a surface that may minimize problems with large slice spacing as well as achieving a 78-88% data compaction rate.

In general, slice interpolation takes advantage of the known structure of the data by filling the space between the slices to create a surface. Certain techniques have the ability to create transitions between slices containing multiple, disjoint contours. Approximating curves reduce the size of the representation and produce uniform surface facets independent of the slice density. However, these methods are still a blind fit to a given data set. Even slight changes in the scan slice angle or resolution will result in a completely different surface mesh.
2.1.3 Volumes

The previous slice interpolation methods are also extended to volume representations by triangulating between internal and external contours [BAJA94, LEVE89]. The three-dimensional model lends itself directly to finite element strength and load calculations for implant design or similar applications within the medical industry. Visual analysis systems also use full volume representations as a basis for generating arbitrarily selected cross-sections or slices, independent of the original scanning orientation.

Simply stacking the original segmented voxel data will simulate a solid model. Software packages, such as ANALYZE, provide the user with tools for extracting lengths, areas, volumes and statistical information directly from the data [BELO91, BELO92, ROBB89]. However, in many applications, these volume representations contain more information than may be necessary, resulting in unnecessarily large data storage requirements and prohibiting real-time interaction [POMM96].

2.1.4 Algebraic Polynomials

In contrast to volume models, surface approximations yield more compact representations of homogeneous volumes by fitting data to a specified tolerance. Depending on the implementation, approximating surfaces may also smooth the data. These surface definitions range from simple algebraic polynomials to Fourier series approximations [AKIM78, ATES93, DUNC90, PRAT93, PURC91a, PURC91b, SCHU90, STAI96, VANO78]. Unfortunately, this particular collection of methods uses a single surface equation and the degree of the polynomial span limits the flexibility of the fit. To improve the approximation, higher degree polynomials may be used; unfortunately, this
degree elevation also increases the potential for unwanted undulations within the surface [DUNC90].

To alleviate this problem, Pratt, Goult and Ye suggest using rational polynomials [PRAT93]. The additional flexibility allows the fit of conic sections, surfaces of revolution and singularities. However, the system becomes nonlinear, making it more complex and expensive to solve. For this reason, many researchers have suggested that nonlinear solutions are currently being used for fashion rather than necessity [FARI89, DOKK91].

Other researchers use superquadrics, such as toroids, ellipsoids and cylinders, as initial surface approximations because the shapes closely mimic many anatomical features [BARD94, SCLA94]. A quadrilateral mesh is first generated to a specified spacing and is then deformed to fit the data. This technique has the advantage that it is simple and provides enough flexibility to interpolate the data. The main disadvantage is that the compactness of the surface description is lost and the approximation degenerates to the simple triangulated meshes discussed earlier.

Lim, et al. present a boolean union technique for combining spherical primitives into a single surface [LIM95]. The location of the centers, radial lengths and blending parameters are optimized to create an approximating surface. As a rational formulation, nonlinear methods are required for its solution. This technique differs from the previously presented methods in that it stitches together multiple, individual surfaces into a single representation. This idea defines the general class of surfacing methods known as splines.

Splines overcome the liabilities of single equation formulations by combining patches in a piecewise fashion to represent a surface. Rather than increasing the degree of a single equation, a collection of multiple patches of a
relatively low degree may fit areas of high curvature. There is a wealth of literature dealing with splines for approximation and interpolation, including Ferguson patches, Coon’s patches, Hermite splines and Bezier patches [BAJC89, BAO89, BAO92, FARI90, FOLE86, KOLB95, SCHM86]. In general, these surfaces can be easily subdivided, providing further fitting flexibility, and have specific advantages in applications such as tool path generation for computer controlled milling [BAO89]. However, all of these formulations are not standard on commercially available software and continuity between the individual patches must be explicitly enforced, restricting their fitting flexibility [FARI89, MILR95].

2.1.5 B-Splines

Bezier surfaces, mentioned above, are a subset of a popular and more general class of splines known as basis splines, B-splines for short [HILL90]. B-spline surfaces are defined by three items: a set of control points or control vertices, two knot vectors, and the associated order of the basis functions. The control vertices are an ordered set of 3-D points that are blended by the basis functions to create the parametric surface. A knot vector is a sequence of monotonically increasing numbers, \( u_i \), that define the parameter space of the surface. The basis functions are evaluated recursively by blending the ranges bounding the specified parametric value, \( u \), up to the desired order, \( r \), as shown in Equation 2.1. The polynomial degree of the basis functions is the “order” minus one.

\[
B_{i,r}(u) = \frac{u - u_i}{u_{i+r} - u_i} B_{i+1,r-1}(u) + \frac{u_{i+r} - u}{u_{i+r} - u_{i+1}} B_{i+1,r-1}(u) \tag{2.1}
\]

where

\[
B_{i,1} = 1.0, \text{ and } B_{i+1,1} = 0.0
\]
A B-spline surface patch requires two knot vectors, one for each parametric direction, and a two-dimensional matrix of control vertices, $V_{u,j}$. With these two items, and the order for the bases in each parametric direction, a B-spline surface patch is defined by Equation 2.2. The variables $\delta_u$ and $\delta_v$ designate which parametric span is being evaluated.

$$\vec{s}(u, v) = \sum_i \sum_j \vec{V}_{\delta_u = i, \delta_v = j} B_i(u) B_j(v)$$  \[2.2\]

Stepping the $\delta$ values in each parametric direction creates the collection of surface patches defined by the given knot vectors and matrix of control vertices. Because of the definition of the basis functions, all of the patches within this collection are degree-minus-one continuous at their respective internal boundaries. The only exception to this rule is when a knot value appears more than once in the knot vector. In which case, sharp corners may be modeled easily by adding multiplicities in the appropriate knot vector position.

When modeling complex objects, applying a single collection of patches defined by one matrix of vertices and two knot vectors may be difficult. Therefore, multiple patch collections are created with separate knot vectors and vertex matrices. This provides the flexibility necessary for fitting complex objects; however, the surface continuity between these collections is not automatic. Continuity must be enforced by restricting the placement of the common edge vertices, a process that reduces the fitting flexibility of the boundary patches [MANN95, LIU89].

To increase the flexibility of the B-spline representation, a weight, which pushes or pulls the surface away from or toward the respective vertex, may be associated with each control vertex [MILR95, PIEG87, PIEG89a, PIEG89b, TILL83]. Combining all of these attributes yields the commercially popular non-uniform, rational B-spline (NURB) formulation given in Equation 2.3,
where "non-uniform" reflects the knot vector spacing. An example of a NURB surface is shown in Figure 2.4. The surface is composed of three patches and requires a 6x4 network of control vertices. Because the basis functions in each direction are of order 4 (degree 3) and the knot vectors do not contain multiplicities, the two patch boundaries shown have explicitly maintained curvature continuity.

\[
\tilde{S}(u,v) = \frac{\sum_i \sum_j \tilde{V}_{\delta_u-1,\delta_v-1} w_{\delta_u-1,\delta_v-1} B_i(u) B_j(v)}{\sum_i \sum_j w_{\delta_u-1,\delta_v-1} B_i(u) B_j(v)}
\]

[2.3]
As long as all of the vertex weights have the same sign and the knot vectors do not contain multiplicities, the surface will have order-2 parametric continuity maintained automatically between patches. The NURB surface will be affine invariant, will be contained entirely within the convex hull of the control net and allow local manipulation for data fitting [PIEG87, PIEG89a, PIEG89b, TILL83]. The rational formulation also provides the added capability of representing quadratic curves/surfaces exactly.

B-splines are an industry standard and have been included in the Initial Graphics Exchange System (IGES) format since 1983 [MILR95, PIEG87]. They define a smooth surface and handle cross-patch continuity automatically. The basis functions provide local control of the surface and can be easily subdivided using knot insertion methods for added flexibility [BART87, BOEH80, GOLD90]. The rational form provides further flexibility for deforming the surface. Because the representation is affine invariant, only the vertices need to be transformed to produce the equivalent effect for the entire surface. Since a surface is created using a relatively small number of components, a NURB surface defines a compact representation. For these reasons, NURB surfaces are used to define the models within this project.

2.2 Approximation / Interpolation Solutions

Once a surface representation has been selected, the next step is to determine the method for fitting the surface to a supplied data set. As with surface descriptions, there is a wealth of information covering these methods. Since NURB's were selected to describe the feature-based models presented in this thesis, the remaining sections will focus on those methods which are applicable to B-spline surfaces, i.e. solving for the vertex positions and weights, knot vectors, and data parameterization.
There are numerous computer programs available for interactively creating B-spline surfaces [BARS80, FORR90, LORD88, MILR95, TOPP90, WELC91, WILK88]. Patches can be placed where desired and deformed manually to better fit the data. The main disadvantage of this process is that it requires extensive user interaction to fit the data to within a desired tolerance. Also, deformations based on knot changes are not intuitive. Therefore, this section will discuss numerical methods that either solve for the surface directly or deform a previously defined surface.

2.2.1 Explicit Solutions

If a non-rational B-spline is used, then the positions of the vertices can be determined by solving a linear system of equations. The number of patches defining the surface establishes whether the system is over-determined, under-determined or has full rank relative to the number of data points in the supplied data set. If it is required that the data be interpolated, then a single patch can be placed between neighboring data points [LEE89, VERG89]. This will result in an under-determined system, having to calculate more vertex positions than data points. Another technique sets the number of vertices equal to the number of data points. This creates a linear system with the same number of equations as unknowns; however, full rank requires that each vertex affect the fitting of at least one data point [BARS80, LORD88, WANG90]. Both of these methods will provide an interpolating surface, but with a model that is larger than the original data set.

Because of the large numbers of points provided by today's scanning systems, approximation methods are often applied to produce a compact description of the object represented by the data. Since the number of control vertices is less than the number of data points, the system is over-constrained.
Therefore, a solution is chosen which minimizes a norm of the error between the surface and data. A least squares norm is commonly used because it is well-behaved and has statistical justification (see Section 2.6.1). More importantly for this application, it also provides a straightforward solution [ATES93, DOBS95, HAYE74, MA95, PARK96, ROGE89, SARK91a, SARK91b, VERG89]. For the linear system, \( Ax = b \), the \( l_2 \) norm finds the vertex solution vector, \( x \), which minimizes the squared distance between the data points, \( b \), and a linear combination of the columns of \( A \), the matrix of basis evaluations [DENN96]. Applying the proof given by Dennis and Schnabel, this overdetermined system can be transformed into a linear system by multiplying each side by the transpose of \( A \) as shown below.

\[
A^T (Ax) = A^T b
\]

The system given in Equation 2.4, known as the "normal equations", may be solved by any number of linear techniques. However, directly solving this simplification may produce under/overflows and round off errors in computer computation and may square the conditioning of the matrix [DENN96, PRES95]. Therefore, methods such as the QR decomposition or the Single Value Decomposition (SVD) should be applied to the \( A \) matrix directly [PRES95, SCHW89, WATK91]. The QR method is numerically stable and less expensive than the SVD; however, an SVD solution should be used if \( A \) contains singularities.

These techniques for creating a surface require the solution of a linear system of equations. Depending on the number of vertices (i.e. patches), the computation may be relatively quick. However, the quality of the fit depends directly on the parameterization of the data. In order to construct the basis matrix, a parameter value must be set which represents the point on the surface
which is closest to each data point. Thus, the parameter values associated with each point are crucial to finding an acceptable solution.

2.2.2 Parameterization of Data Points

With a parametric representation, a pair of parametric values uniquely defines each point on the surface. To fit a parametric surface to a data set, each data point must be associated with a point on the surface (i.e. a specific parametric pair). Therefore, the parametric selection directly affects the quality of the surface fit.

The structure of the data set may suggest how the parameterization should take place. For example, if the data is a rectangular grid of points, the parameters are set by dividing the given knot spans by the number of points in each direction [FOLE86]. This is known as the uniform parameterization because it assumes that the data points are evenly distributed about the object's surface [SARK91a, VERG89]. This is a very simple method, but it contradicts the tenet of placing more data points in areas of high curvature.

Another technique sets the parameters based on relative arc length for curve fitting. Assuming that the connectivity between points is known, the chord length method traverses the points to calculate the piecewise linear distance between the first and last data point. Knowing the parametric span for the curve, the parameter value for each point is set relative to its chord length from the first point. This technique has been applied in 3-D surface fitting by Schmitt, et al. using the position of a data point relative to a previously defined surface patch [SCHM86].

Using an analogy presented by Lee, the chord length method assigns the parametric values as if a car was moving along the path at a constant speed [LEE89]. The parameterization is the same whether the points were in a
straight line or defining a highly curved path. This disregards the common fact that a car normally slows in high curvature turns to account for the increased centripetal acceleration. Therefore, a better method would constrict the spacing of the knots in areas of high curvature. The "centripetal" parameterization, derived by setting $p=0.5$ in Equation 2.5, improves the previous solution using the data point $(P)$ Euclidean distances [LEE89].

$$u_i = u_{i-1} + \frac{\|\bar{p}_i - \bar{p}_{i-1}\|^p}{\sum_{j=1}^{N_{\text{data}}} (\|\bar{p}_j - \bar{p}_{j-1}\|^p)}$$  \hspace{1cm} [2.5]

If $p=0$, the uniform distribution is acquired and setting $p=1$ results in the chord length parameterization. Thus, the centripetal method is a blend of the uniform and chord length methods. Any other numerical value for $p$ shifts the parameterization toward the respective extreme [LEE89, MA95].

2.3 Adjustment Techniques

Once the data has been parameterized, the surface description can be resolved using the techniques presented in the previous section. However, this initial surface may not adequately fit the data. Therefore, techniques have been derived to adjust one or more of the surface components, such as vertex position, weight or data parameterization, to better fit the data. This section will present some of these methods as found in the literature.

2.3.1 Iterative Addition/Subtraction

The main philosophy behind surface fitting is to provide a compact description while approximating the data to within a specified tolerance. Since the number of patches to be fit is user specified, the linear system solution techniques presented previously will not usually result in an optimal representation. Either the surface will not fit the data to the given tolerance, or
there are more patches in the representation than necessary. Hsieh addresses this problem by iteratively decreasing the order of the surface until the data is within tolerance [HSIE93]. Obviously, the process ends when the surface reaches a first order approximation, completely losing its compact nature.

A more common technique iteratively adds or subtracts patches to find an “optimal” representation [PARK96, SARK91a, VERG89]. The errors (Euclidean norm) between the data points and their respective points on the surface are calculated. If the largest error falls out of the tolerance, another knot span is added and the linear system is resolved. If all of the points are within tolerance initially, one knot span is removed and the process iterates.

Either technique requires a reparameterization of the data to reflect this change in one of the parametric knot spans. If there are a large number of data points or surface spans, the process may be computationally expensive. More importantly, for the B-spline formulations being used, an additional span does not add just a single patch but a series of patches along the other parametric direction. Approximating a large data set to a small tolerance with these methods produces a nearly equally as large, somewhat arbitrarily positioned collection of control vertices.

2.3.2 Adaptive Subdivision

In order to minimize the size of models, several authors have proposed adaptive techniques for subdividing the surface locally in areas where the tolerance is not met. The idea is similar to the knot/vertex addition methods given earlier and two techniques presented by Boehm and Goldman for inserting knots into a currently defined knot vector [BOEH80, GOLD90]. The exception is that only those patches that are out of tolerance are subdivided.
Schmitt, et al. [SCHM86], give a good example of adaptive subdivision techniques. An initial coarse mesh of Bezier patches is fit to a data set. Each patch is then systematically tested to see whether the patch approximates its associated data to within a specified tolerance level. If all of the errors for a patch fall below this level, then the patch is accepted and nothing further is done. However, if the patch does not meet the tolerance, it is subdivided at the mid parametric value in each direction, generating four smaller patches. This process continues until all of the errors fall within the specified tolerance. Similar versions of this technique have been implemented by other researchers because it provides a compact representation [BAO92, CHOU92].

The knot insertion technique introduced in Section 2.3.1 may be used to subdivide a general B-spline patch. However, for generalized B-spline formulations, the division cascades through all of the surface patches defined by the span. Therefore, Forsey and Bartels have developed an overlay method for subdividing a single B-spline patch without affecting neighboring patches. The patch is redefined into 16 smaller patches by adding three knots in each parametric span defining the patch. The neighboring patches are evaluated using the original knot vectors. The OSLO algorithm is then applied to calculate the 49 new vertices, a 7x7 mesh, for the 16 subdivided patches which represents the exact same surface as the original patch [BART87, GOLD90, LYCH86]. These new patches are evaluated using the refined knot vectors. To maintain the continuity at the boundaries with the unrefined neighboring patches, only the central vertex of the 49 is allowed to move to improve the fit.

Initially, this may seem like a large addition to the surface; however, only the change in the moveable vertex need be recorded to create the refined surface patch. The mid-vertex offset is multiplied by the associated refined
basis functions and is added to the original description for the patch. Subsequent divisions of these patches are made in a hierarchical manner, building on the previous refinement, until the data falls within tolerance [DOBS95, FORS90, SZEL90, SZEL94a].

2.3.3 Closest Point Calculations

Once an initial surface has been created or a patch subdivided, the control vertices must be adjusted to better fit the data. The first step in this process is to determine the parameter pairs for the closest point on the surface to each data point. For the triangulated surfaces presented earlier, the simple geometric representation allows for a quick closed form solution for the closest points [BORI48, DICK14, HAUS90, MOSH94]. The same also holds true for parallelepiped volumes, such as free-form deformation matrices (the parametric directions are independent) and curves, which simplify into a single parametric polynomial [HSU92, ODES93, SZEL94b].

Because B-spline surfaces are composed of a polynomial product, direct solution of the closest point parameters is difficult. For this reason, researchers use optimization methods for calculating the closest point on a surface to a data point [BESL92, CHOI93, DOBS95, LAUR93, MA95, MILR95, PARK96, ROGE89, SARK91a, SARK91b, SCHM86]. Since most optimization methods are only capable of finding a local minimum, a "good" starting value is required for the iteration. A common method evaluates the surface at each knot value, or some parametric step, and determines which mesh point is closest to each data point. The initial value for the optimization is set to the parametric value of the closest mesh point [BARD94, BESL92, CHEN94, DOBS95, HAUS90, LORD88, MOSH94, PELI89].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
2.3.4 Iterative Adjustment Techniques

With an adjusted parameterization of the data, a new surface can be calculated using the linear system methods presented earlier. This procedure is iterated until the data is fit to within tolerance or there is little change in the error. The main drawback of this technique is that the solution of the linear system may be computationally expensive, making iterative solutions unrealistic. Therefore, adjustment techniques are applied to deform a currently existing surface.

Marilyn Lord uses the error vectors resulting from the closest point calculations to move the vertices [LORD88]. The process starts by coinciding the control vertices with the data points, yielding a rough approximation to the surface. If the database is very dense, with more points than desired model vertices, some data points may be dropped without significantly affecting the error [LORD88, WANG90]. Starting with the first control vertex, the closest point on the surface is found and the vertex is offset by the vector from the surface to the data point. The closest point offset is iteratively applied to this control point until the error between the data and surface is within tolerance. This process is then applied to each vertex separately until all of the vertices have been moved. Since the subsequent movements may affect previous fits, the entire process may be repeated as necessary. The process is simple and not computationally expensive; however, the quality of the surface depends on the initial selection of the data subset.

In a similar manner, other researchers try to move specific points on the surface to a location in space [DOBS95, GEOR92, HSU92, WELC91]. The movements may be performed interactively or automatically using the closest point errors to control the movements. The vertices are then calculated using an
under-determined linear system to effect the required surface movements. These deformations can be used in conjunction with knowledge of the object in order to perform changes based on features within the surface [DOBS95, KAPO98, MOSH94, SCLA94].

Because the B-spline formulation supports local manipulation, the point movement method affects only a small section of the surface neighboring the point in question. This may be advantageous in some applications; however, the deformation may cause "spiking" in the surface. To broaden the effect of this process, the surface may be embedded within a free-form deformation (FFD) matrix to perform the modification [BARD94, FORS90, GEE93, HSU92, LAMO94, SEDE86, SZEL90]. Deformations of the coarse FFD matrix result in a smooth, broader deformation of the embedded surface, as shown in Figure 2.5. The degree of smoothing is directly dependent on the location of the vertices within the FFD matrix, their location relative to the deformation, the coarseness of the FFD matrix and the degree of the basis functions within the parametric solid [LAMO94].

Figure 2.5 Example of a free-form deformation on a cylinder.
The iterative methods discussed earlier attempt to minimize the error between the surface and the data set. Therefore, researchers have proposed applying explicit optimization algorithms to the surface parameters, including the vertex position and weights, closest point parameters, and knot sequence [DOBS95, LAUR93, PARK96]. If multiple, individual patches are to be combined to form a surface, the optimization can force levels of continuity between the distinct patches [MILR95]. These algorithms may also be applied to the polynomial representations presented earlier or free-form deformation grids to deform point meshes [BARD94, DUNC90, LIM95, STAI96, SZEL94b]. Optimization methods are a more complex solution than the iterative methods given earlier; however, depending on the technique, they may converge quadratically to a minimal solution. For this reason, optimization methods are used to deform the surface models to fit the data in the proposed feature-based modeling technique.

2.4 Preliminary Modeling Systems

All of the techniques mentioned thus far fit an arbitrary collection of patches to a data set. Rescanning the object in a slightly different orientation or simply rearranging a point cloud generally results in two completely different patch arrangements or models. Thus, there is no coherent model between fits other than the fact that both approximate the data to the same tolerance. No advantage is taken of the fact that the general shape may be known prior to fitting the data. For example, medical structures have a large range of "normality", only differing in individual shape details [BAJC89, GEE93, SCLA94]. Therefore, a logical starting point for a fitting process should be a smooth surface model having this generic shape. The problem is to deform the existing surface such that features within model match the associated features in
the data while maintaining the inherent shape. This is akin to morphing one image into another. Specific points in the first image should migrate to the same point in a second image [BEIE92]. This keeps the patch relationship consistent between fits, providing a basis for comparison between data sets and extracting measures, the most important challenge in medicine [ROBB89].

Szeliski and Lavallee present a technique for morphing one point cloud to match another [SZEL94b]. The process uses only global transformations, which limits the quality of the fit. To fit surfaces to a small tolerance, a method must be able to perform localized deformations of the model. Also, the point cloud model used by Szeliski and Lavallee is as large as the data set being fit; no data compaction is provided. Other researchers have developed stacked slice models that are simple to edit interactively and provide a smaller representation than the data cloud [BOON89, GEE93, HOUS92, TORR92]. However, the models are not surface representations and the deformations may cause continuity problems between the slices. It is also more computationally expensive to quantify lengths, surface areas and volumes from stacked slice models [FABE88].

Discretized surfaces, such as triangulated meshes, are also used as preliminary models [BARD94, FABE88, HAUS90, TOPP90]. Since each component is a geometric form, the closest point calculations are a simple, closed form solution. The surface also provides a basis for quantifying 3-D values, such as lengths, areas and volumes. However, discretized surfaces are not compact representations; sometimes resulting in files larger than the original data set. Full volume models include interior as well as exterior features and provide a basis for segmentation of voxel data; yet, these
representations are more complex than surface models, making them much harder to fit [AYAC96, BAJCS89, FABE88, KAPO89, MOSH94, WHIT94].

2.5 Registration

The first task in fitting an existing model to a particular data set is to bring the two elements into close proximity and align them prior to performing any detailed fitting procedure. Position, orientation and possibly scale should be optimized such that the defining features of the model and data coincide. The method for registering the model to the data depends on the structure of the data set, the surface representation selected, and the allowable complexity of solution. Interactive methods provide the user with multiple views of the scene to manually adjust position, rotation and scale [HAUS90, PELI89, SIU96]. The resulting alignment may be accurate, but requires intense interaction of a skilled operator, making it undesirable for fitting large numbers of data sets.

2.5.1 Landmarks

Anatomical features or landmarks, such as the mid-patellar tendon for below-knee prosthetics [HOUS92] or the inter-hemispherical fissure in brain modeling [KAP90], have successfully assisted in identifying the proper orientation [ASHL93, DOBS95, HAUS90, HOUS92, KAPO89, LAVA96a, PELI89, SIDL89, VAND93]. These landmarks may be internal, existing on the surface of the model, or external, including fixturing devices or block markers [PELI89, VAND93]. Internal landmarks may not be distinguishable across all scanning devices; or, if available, the methods for automatically locating them within a data set are complex and computationally expensive. The sharp corners of external landmarks, on the other hand, provide easily distinguishable
boundaries; however, their placement within subsequent scans is less reproducible.

If at least three specific point-to-point correspondences are available between the model and data set, then the transformation for positioning and orienting the model can be calculated explicitly and efficiently [ARUN87, FABE88, HORN91a, HORN91b, HUAN86, ZHUA97, ZUK96]. The effectiveness of the registration depends directly on the accuracy of the landmark locations [TOEN90]. Because this subjective identification is susceptible to error, more than three landmarks are commonly used in a least squares solution to the linear system.

2.5.2 Moments

To minimize the need for expensive feature extraction algorithms, moment analysis methods, which take advantage of the common character of the model and data set, are used [BEER84, LEU91, SING93, TORB84]. No prior knowledge of landmarks or shape is required to calculate the alignment transformations.

Matching centers of mass of the model and data set has shown to be a reasonably effective way of positioning the model to the data [BAJC89, CYGA85, FABE88, GEE93, HAUS90, HOUS92, HUAN86, KAPO89, LEE89, LO90, LORD94, MOSH94, TOEN90, VAND93]. Orienting the model, on the other hand, has proven to be a more complex task. Moment tensor analysis provides an acceptably accurate method for scaling and positioning two objects that are affine maps of one another [BORI68, CYGA85, FABE88, HORN91a, LEE90, LO89, LO90]. This technique works well in the field of motion tracking where the objects are not changing except in scale, position and orientation. In anatomical modeling however, the model and data set are rarely
exact affine transformations of one another; therefore moment tensor analysis is not recommended [FABE88].

Principal axes alignment yields a more accurate registration than the tensor product solution when positioning inexact transformations [FABE88]. Symmetry within the data may result in large errors in the orientation since no unique set of axes exists [ALPE90, BAJC89, DOBS95, FABE88, GEE93, HAUS90, HU62, KAPO89, LEE89, LORD94, MOSH94, TOEN90, VAND93, WELL67]. This method also assumes that all scaling occurs along the principal axes, which is rarely the case. Since the output of the process is a set of three orthogonal axes, there are eight possible complementary vector bases that establish the orientation. Additional cues are required in order to set and improve the alignment.

2.5.3 Optimization

Landmark and moment alignments provide a rough registration of a model to data and are a good choice as initial solutions for minimization routines. The solution may be formulated using screw axes or dual quaternions, but the most common methods use sequential axis rotations [BESL92, BOUR88, HORN91a, HORN91b, LAVA96a, ZHUA97]. Minimization routines may be applied directly to solve for the optimal position, orientation and scale parameters [CHOI93, DOBS95, HAUS90, HORN91b, HOUS92, LO89, PELI89, SIDL89, SZEL94b, ZUK96]. These methods may be complex, but their convergence rate ranges from linear to quadratic, depending on the method. Because most optimization routines are local minimizers, interactive steering may be necessary to force the solution toward the proper minimum [PELI89].
To simplify the alignment process, Huang, et al. suggest solving for each angle separately, since a rotation angle can be calculated explicitly for the planar case [HUAN86]. The method projects the closest point errors for the data set onto the plane normal to the current axis. The routine does not optimize position; it simply uses the center of mass matching. In a later paper, Huang, et al. present a second routine which approximates the nonlinear least squares solution with the linear system given in Equation 2.6 [HUAN96]. The closest points are used in an iterative solution for the position and orientation only. Each iteration within these techniques is quick and simple to compute, but the convergence rate of the solution is yet unknown.

\[
[T]^{-1} = ([CP]^T [D])([D]^T [D])^{-1}
\]  

where

- \([T]\) = 4x4 transformation matrix
- \([D]\) = 4x\(N_{\text{data}}\) matrix containing all of the data points
- \([CP]\) = 4x\(N_{\text{data}}\) matrix of closest points to each data point in \([D]\)

Articulated models present a special problem in which different components of the model should be aligned separately. The variable position of toes, ankles or fingers, for example, may prevent a static model from aligning correctly. Additionally, disease or trauma may bend bones beyond their generic shape such that aligning the proximal and distal halves separately gives the smallest registration error [HAUS90].

Although each of the methods presented has advantages, no single technique is clearly superior. Therefore, a combination of these methods should be used for a robust registration algorithm [ZUK96]. If their acquisition is not expensive, landmarks should be used to overcome symmetry problems and aid alignment. Principal axes and center of mass positioning offer a good
initial registration for subsequent optimization. Finally, user interaction may be required to insure that the desired solution is achieved.

2.6 Nonlinear Systems (Optimization)

The solution of nonlinear systems appears in all components of the fitting process as has been explained earlier. Closest point parameterization, registration and surface fitting may all be solved using nonlinear optimization methods. Therefore, these algorithms are reviewed in the following sections.

There are three general types of optimization routines: those that use function evaluations alone, those that use first derivative evaluations, and those that use both first and second derivatives. Direct search methods use only function evaluations; an example of which is the well-known Powell’s method. These methods excel when the function gradients are unavailable or are expensive to calculate; however, they are not recommended if the function evaluation is expensive to compute.

If the derivatives are known analytically, direct search techniques are as much as \( n \)-times more computationally expensive (\( n \) being the number of variables) than gradient-based methods [LAUR93, SPANG62]. For this reason, the remainder of the section will focus on routines that make use of the analytical derivatives.

2.6.1 Cost Function Norms

For data fitting, optimization routines are commonly used to find the surface parameters that minimize the error between the surface and data points. To apply these routines, the individual data errors must be combined into a single cost function such that the data is “best” fit. There are many different
norms that are appropriate for this task; however, only the three more commonly known norms ($l_1$, $l_2$, and $l_\infty$) will be discussed.

The $l_1$ norm, also known as the least absolute residual norm, simply sums the absolute value of the errors as shown in Equation 2.7 [DENN96]. Because of its insensitivity to outliers, the $l_1$ norm should be used when there is little control over error during data collection [TARA87].

$$l_1 \text{ Norm} = \sum_i |x_i|$$

[2.7]

In contrast, the $l_\infty$ norm, Equation 2.8, requires a strict control on the error, allowing only for round off errors for example [TARA87]. Since both formulations are not analytic, finite difference methods are required to approximate their derivatives [CHOI93].

$$l_\infty \text{ Norm} = \text{Max} |x_i|$$

[2.8]

As with the other two norms, the $l_2$ norm has its basis in probability densities (Equation 2.9). The least squares norm assumes that the errors within the data maintain a Gaussian distribution, a situation that rarely occurs in practice [BOUR88, CHOI93, TARA87]. Barring this statistical limitation, the least squares norm has become the standard for data fitting because it is simple, continuous and twice differentiable [BOUR88, CHOI93, TARA87]. The norm tends to smooth errors within the data; however, it is still sensitive to a small number of large outliers [DIXO72, TARA87].

$$l_2 \text{ Norm} = \sum_i (x_i)^2$$

[2.9]

To determine which formulation is better suited for registration, Bourdet tested the $l_\infty$ and least square norms on the alignment of common geometric....

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
shapes [BOUR88]. The infinity norm usually resulted in a 15% greater decrease in the cost function as compared to the $l_2$ norm. However, as the number of data points increased past 20, the $l_2$ norm approached the $l_\infty$ norm, negating its advantage. Also, the solution using the $l_2$ norm was less expensive to calculate, reaching a minimum in nearly half the time. Dennis and Schnabel state that the choice of norm is unlikely to affect the performance of the algorithm [DENN96]. Therefore, the selection should be made as a matter of convenience. For these reasons, the least squares norm is used exclusively in the proposed algorithm.

2.6.2 Conjugate Gradient Methods

The gradient of a function yields the vector of greatest functional change from the current position. Logic would dictate that methods iteratively searching along the negative of this vector would minimize the function efficiently. However, the convergence is generally very slow, even on simple quadratic forms with a long valley sloped toward the minimum [PRES95]. This steepest descent technique produces search directions which are perpendicular only to the previous vector. To keep from "spoiling" the previous result, the new search direction should be conjugate to the last direction as well as all of the previously calculated directions [PRES95]. This results in the general class of conjugate gradient minimization routines.

The first iteration uses the negative of the gradient as the search direction. To enforce the conjugacy, the subsequent directions are calculated using the current and previous gradients, the previous search direction and the second derivative matrix (Hessian matrix) of the function at the current position. Press, et al. present a derivation of a search direction calculation which does not require the Hessian matrix [PRES95]. This simplification
yields two well-known versions of conjugate gradient updates: Fletcher Reeves (Equation 2.10) and Polak-Ribiere (Equation 2.11) [HU91, LAUR93, LIU91, PRES95]. Of the two, the Fletcher-Reeves update is the simplest to calculate, but the Polak-Ribiere technique is more robust on non-quadratic equations.

\[ s_{k+1} = \frac{\bar{g}_k^T \bar{g}_{k+1}}{\bar{g}_k^T \bar{g}_k} \]  \[ \bar{g}_{k+1} = \frac{\bar{g}_k^T (\bar{g}_{k+1} - \bar{g}_k)}{\bar{g}_k^T \bar{g}_k} \]  \[ 2.10 \]  \[ 2.11 \]

When a search direction has been selected, the next solution is determined by minimizing the function along this vector. Univariate optimization routines, such as bisection or Golden section searches, are used for this purpose. Unfortunately, conjugate gradient techniques are very sensitive to this line search, requiring exact minimums for best convergence [DENN96, LIU91]. All of the variables should be of the same scale (order of magnitude) and the search directions may require a reset to the negative gradient to improve convergence. At best, conjugate gradient methods have super-linear convergence, but are comparable and utilize less storage than the more powerful Newton-type optimizations for large-scale problems.

2.6.3 Newton-Type Methods

Mathematically, the minimum of a function occurs at a point where the gradient of a function is zero. Thus, root-finding methods, such as the Newton-Raphson technique, are applied to calculate the minimum of a function by finding the point where the gradient diminishes. Derived by either applying a first order Taylor series expansion or simplification of Newton's theorem on the
gradient of the function, the general form of Newton’s method for a scalar function minimization is given in Equation 2.12 [DENN96].

\[
f''(\bar{x}_k) \bar{s}_k = -f'(\bar{x}_k)
\]

\[
\bar{x}_{k+1} = \bar{x}_k + \bar{s}_k
\]

[2.12]

Application of this technique transforms the original nonlinear problem into the much simpler solution of a system of linear equations. To prevent the iteration from converging to a maximum or a saddle point (which both have zero gradients), Newton’s method requires that the Hessian matrix is positive definite (all positive eigenvalues).

The nonlinear least squares problem is given in Equation 2.13, where \( R(x) \) is the residual vector function of \( m \) errors in \( n \) variables [DENN96].

\[
f(\bar{x}) = \frac{1}{2} \sum_{i=1}^{m} \left( \tilde{s}_i(\bar{x}) - \bar{p}_i \right)^2 = \frac{1}{2} \sum_{i=1}^{m} \tilde{R}_i(\bar{x})^2
\]

\[
= \frac{1}{2} \sum_{i=1}^{m} \tilde{R}_i^T(\bar{x}) \tilde{R}_i(\bar{x})
\]

[2.13]

Using the first and second order gradients of \( f(x) \) given in Equations 2.14 and 2.15, respectively, the solution step is found by solving the linear system in Equation 2.16. If this method is implemented as presented, the error will decrease quadratically between steps.

\[
J_q(\bar{x}) = \frac{\partial r_i(\bar{x})}{\partial x_j}
\]

[2.14]

\[
[G(\bar{x})] = \sum_{i=1}^{m} r_i(\bar{x}) \ast \nabla^2 r_i(\bar{x})
\]

[2.15]

\[
[J(\bar{x})^T J(\bar{x}) + G(\bar{x})] \bar{s}_k = -J(\bar{x})^T \bar{R}(\bar{x})
\]

[2.16]
The problem is that the second gradient matrix, $G(x)$, may be unavailable or expensive to calculate using numerical approximation methods [DENN96, DIX072]. If each residual component equals zero at the minimum, $R(x^*)=0$, then the $G(x)$ matrix may be dropped without loss of quadratic convergence [DENN96].

Zero residuals occur in data fitting when the surface interpolates the data; however, in most cases, the surface can only approximate the data. Thus the residual at the minimum will fall into one of two categories: small or large [DENN96]. The names connote a dependence on the numerical size of the approximation errors for classification; however, a simple scale factor would transform any large residual problem into a small residual one. The true determination compares the effect of $G(x)$ relative to the $J(x)^T J(x)$ component. If the matrix norm of $G(x)$ is sufficiently smaller than the norm of the Jacobian product, then the system is a small residual problem and $G(x)$ may be disregarded. Otherwise, the system has a large residual and the $G(x)$ matrix must be used in order to assure convergence [DIX072].

Dropping the second derivative component from the full Newton's method generates the Gauss-Newton class of optimization techniques. These methods include the damped, Gauss-Newton method, which performs a univariate search along the calculated direction, and the Levenberg-Marquardt method, which applies a trust region approximation to the disregarded $G(x)$ matrix [DENN96, FORS97, GILL78, GULL97, MARQ63]. These methods are best applied to zero or small residual problems where the convergence ranges from quadratic (zero residual) to super linear (small residual). For large residual problems, the second derivative matrices of the residual equations must be included. If not known analytically, the derivative may be approximated by
finite difference methods or by iterative secant method approximations [COLE84].

Newton methods are very effective when the second derivatives are known analytically and linear searches are expensive [DENN96, LAUR93]. However, these methods may become tedious as the number of variables increase since it must solve a linear system at each iteration, $O(N^3)$ [DIXO72, LAUR93, SCHN82, SZEL94b]. Methods like the Cholesky decomposition may be used which take advantage of the required, positive definite nature of the second derivative matrix [DIXO72, SPAN62]. Gauss-Jordan or successive relaxation methods may also be implemented [GERA89]; however, Tarantola states that a conjugate gradient optimization is more efficient than a Newton’s method using an iterative solution [TARA87]. Near the minimum, exact solutions to the linear systems are not required for convergence. Therefore, Szeliski suggests using a single iteration of a conjugate gradient technique to solve the linear system for the next Newton step [SZEL94b].

2.6.4 Mixed Systems

In many applications, the optimized variables are both linear and nonlinear. For example, the vertices in a B-spline fitting process may be solved by linear least squares while the nonlinear closest point parameters and vertex weights can not. Golub and Pereya have showed that the linear and nonlinear variables can be solved independently in an iterative scheme [GOLU73]. First, the linear variables are determined by linear least squares; then the nonlinear variables are optimized. The two steps are iterated until convergence. Separating the solution usually requires less time and fewer function evaluations than standard nonlinear least squares code and no starting values are required for the linear variables. Laurent shows that the linear B-spline
vertex positions are a source of ill-conditioning in surface fitting [LAUR93]. Solving for the vertices separately improves the convergence rate of vertex weight, closest point parameter and knot vector optimization.

2.7 Summary

Numerous surface representations and fitting algorithms are presented in this chapter. The feature-based fitting algorithm to be discussed in the next chapter incorporates a number of these topics into its formulation. B-spline models are used exclusively because of the inherent continuity control, localized deformation properties and affine invariance. Since multiple data sets with a common shape are being fit, the feature-based algorithm fits a previously defined surface model to a supplied data set. The model must be registered with the data to place the two objects in a common orientation. Next, the model is deformed using point deformation tools to move features on the model to corresponding features in the data. Finally, the model is fit to tolerance using optimization routines and hierarchical refinement, if necessary. Specific information about the included algorithms and the innovative additions are discussed in the following chapter.
Chapter 3  Feature-Based Fitting Methodology

As shown in the previous chapter, there are numerous methods for fitting a surface to an acquired data set. Most commercial scanners now come with some sort of bundled surfacing software. However, the surfaces provided by these methods are completely arbitrary. No advantage is taken in applications where the geometric shape of the object being fit is known a priori; a possible source of computational inefficiency when fitting multiple scans of the same shape. The proposed method builds a surface model with this generic, inherent shape. Additionally, specific features are flagged for use in roughly fitting the data and extracting required information. Once fit to within a specified tolerance, the resulting surface provides a compact, consistently sized representation. Since the important measures are incorporated within the model through the feature areas, information extraction and model-to-model comparisons are simplified.

The feature-based modeling technique is separated into four basic tasks: model design, initial alignment and scaling, rough fitting, and tolerance fitting. In model design, the key features of interest within the data sets are incorporated into the definition of the generic model. The fitting process starts by appropriately aligning and scaling this model to the data. With the model situated, a first fit of the data is accomplished using the inherent model features. As a final step, the model is refined and tweaked by optimizing the geometry to meet a desired tolerance. The process has both 2-D curve and 3-D surface analogues, but the procedure is presented in a general three-dimensional surface
formulation. For visual clarity, accompanying explanatory figures will reflect a two-dimensional curve implementation.

The first section describes the general process by which the base or "generic" models may be constructed. The next three sections outline the intermediate steps in the fitting algorithm. Section 3.2 presents the various registration routines that align and scale the model to a supplied data set. Once aligned, the model is deformed using the inherent features defined in the model's creation (Section 3.3). Next, Section 3.4 describes the B-spline control vertex optimization to fit the data to a supplied tolerance. If the optimization fails to meet the tolerance supplied, the model is subdivided using the hierarchical refinement process discussed in Section 3.5. To demonstrate the fitting capability of the algorithm, the feature-based fitting process is implemented using both 2-D curve and 3-D surface models (Section 3.6). This section presents a published pilot study using 2-D facial silhouettes to test the initial concepts of the algorithm [DOBS95]. Next, a model fit to laser scans of a human foot shows the extrapolation of the algorithm to three-dimensional surfaces. Accuracy, repeatability and sensitivity testing of the algorithm are discussed in Chapter 4.

3.1 Model Construction

Before the process can be applied for data fitting, a model must be constructed. The surface should reflect the generic shape of the object being fit, providing enough detail and fitting flexibility to account for individual deviations. Defining features on the surface of the object are selected to control the deformation and provide tools for feature extraction.

For this implementation, cubic non-uniform rational B-splines are used to define the model. As a parametric formulation, the NURB representation
provides a simple basis for embedding features into the model's geometric structure and enforces an order to the topology of its surface. In other words, the position of pertinent geometric features is fixed relative to adjacent patches of the model. As the model is deformed to fit a data set, a correspondence is established between the characteristic regions within the data and the features upon which the model is based. As a result, important measures within the data are readily extracted from the fitted model since they are inherent to its definition.

The base model may be constructed using any technique available; one of the techniques mentioned in Chapter 2 or commercial software for example. For simplicity, the models presented herein are constructed using a sample data set that adequately represents the generic object shape without containing extreme deviations from a norm. Once selected, any one of the methods presented for surface construction is viable for fitting a NURB surface to the data set. Interactive methods allow the greatest freedom for applying patches to the surface, but this process is extremely user intensive. The preferred method solves for the vertex positions using a least squares linear system approximation. The number of patches to fit in each parametric direction and the initial weight values (unity) are input to the system. The data is parameterized using either the chord length or the centripetal techniques discussed earlier. If the initial solution does not adequately represent the shape, then the surface may be adjusted using the deformation techniques discussed previously or a new surface may be constructed with a different number of patches or data parameterization.

Once a suitable model has been developed, characteristic regions on the surface are flagged for use by the feature-based deformation tools. First,
parametric points on the surface are selected which correspond to specific points of interest in the shape. For example, the tip of the nose and the cusp of the mouth are defining features on the 2-D facial profile shown in Figure 3.1.

To associate data points with a specific feature, a parametric range is set which bounds each feature point. If a closest point parametric value of a data point falls within a feature range, then that data point becomes associated with the respective feature. The closest point error vector for that point is subsequently used in the feature deformation routine to adjust the position of the characteristic point. This process will be discussed in detail in Section 3.3.
3.2 Model Registration

As mentioned previously, numerous researchers have studied the problem of registering one object to another. The algorithm should minimize the need for user interaction and be relatively insensitive to data density and structure (point cloud or slices). No clearly superior method exists for aligning the model to the data. Therefore, a combination of various methods provides the soundest approach.

3.2.1 Initial Alignment

The first step in the feature-based fitting algorithm consists of an initial alignment using landmarks, if available, and principal moments. If not provided explicitly with a data set, landmark extraction from the data would be far too expensive. Therefore, center of mass positioning and principal axes alignment are used. The method for calculating the moments depends on the structure of the supplied data. If the data are ordered, such as slices or triangulated points, then the volume can be approximated as a collection of tetrahedra referenced to a single point. The center of mass of the data is easily calculated using the known center of mass and volume of each tetrahedron. The method presented by Lien and Kajiya is then applied which sums the moments of the individual tetrahedron volumes to determine the moments of inertia of the data set [LIEN84]. This same technique is applied to a discretized approximation of the surface model to calculate its moments.

If no order is supplied, then the data set is assumed to be a simple cloud of points. The center of mass is taken as the average value of all the point positions. Since the volume can not be approximated without applying a costly topology routine (Delauney triangulation, for example), each data point is assumed to be a point mass. Presented by Patrick Lord, this technique assumes
that the data are uniformly spaced about the object's surface [LORD94]. Density fluctuations within the data may result in an inaccurate calculation of the principal axes, causing errors in the alignment.

The principal axes are the eigenvectors of the moment tensor. The axes and associated principal moments are calculated using a Jacobi iteration [PRES95]. The moments are ordered into decreasing value so that they may be matched with corresponding moments of the model. The first two axes are assumed to be the z and y axes, respectively. The third axis, the x-axis, is tested to make sure that it forms a right-hand orthogonal basis with the other two axes. Using the centers of mass and corresponding unit axes for the data \((U_{data}, V_{data}, W_{data})\) and the model \((U_{model}, V_{model}, W_{model})\), the model is initially registered by the transformation in Equation 3.1.

\[
[T] = [T_{COM_{data}}] \begin{bmatrix}
U_{data} & \rightarrow & 0 & 0 & 0
0 & \rightarrow & S_{c_x} & 0 & 0
0 & \rightarrow & 0 & S_{c_y} & 0
0 & \rightarrow & 0 & 0 & S_{c_z}
\end{bmatrix} [T_{COM_{model}}] [3.1]
\]

The scale values, \(S_{c_x}\), are determined by the max/min size ratios of the model and data along each of their respective axes.

### 3.2.2 Closest Point Calculations

To evaluate the accuracy of the initial alignment, the closest points on the surface to each data point must be determined. Since a closed form solution of the parameters is difficult, an optimization routine is used to find the point on the surface that minimizes the distance to a data point. This is shown by the cost function in Equation 3.2.

\[
Cost(u,v) = N_{data} \sum_{i=1}^{N_{data}} \left( \vec{S}(u_i,v_i) - \vec{P}_i \right) \cdot \left( \vec{S}(u_i,v_i) - \vec{P}_i \right) = (S_x(u_i,v_i) - P_{ix})^2 + (S_y(u_i,v_i) - P_{iy})^2 + (S_z(u_i,v_i) - P_{iz})^2 [3.2]
\]
Because the equation is continuous, twice differentiable and is a function of only two variables, a Newton optimization is ideal for the solution. Denoting the second derivative Hessian matrix as \( H(u,v) \) and the gradient as \( g(u,v) \), Newton's method computes the next step in the iteration by Equation 3.3 (\( k \) denotes the iteration step).

\[
\begin{bmatrix}
  u_{k+1} \\
  v_{k+1}
\end{bmatrix} =
\begin{bmatrix}
  u_k \\
  v_k
\end{bmatrix} - [H_k(u,v)]^{-1} g_k(u,v)
\]  
[3.3]

To insure that the solution is a minimum, the Hessian matrix must be positive definite. If the eigenvalues of the matrix are not both greater than zero, then the solution may proceed to a maximum or saddle point. To prevent this, Dennis and Schnabel suggest forcing the matrix to be positive definite by adding a number slightly greater than the most negative eigenvalue to the diagonal elements [DENN96].

Because the optimization does not find a global minimum, a "good" initial value must be supplied such that the algorithm converges to the proper local minimum. When no previous closest point solution is available, the surface is discretized at a set parametric increment and the parameter values of the closest mesh point are used. The optimization of equation 3.2 is performed on each data point separately as needed.

3.2.3 Optimized Registration

The center of mass positioning and principal axes alignment place the model in close proximity to the data, suitable for a local optimization method. Since the goal of the optimization is to minimize the distances of all of the data points together, a least squares norm is utilized to combine the numerous errors into a single cost function (see Equation 3.4). The transformation matrix \([T]\), is a function of the position, orientation and scale of the model. Again, a full
Newton's method may be applied since the function is twice differentiable and the number of variables is small (a total of nine). Because the rotation angles at the minimum may differ by as much as 45 degrees from the initial values determined by the principal axes alignment, the closest point parameters are reinitialized to their closest mesh points at each step in the optimization. This ensures that the minimums determined by the closest point optimization are global and not tied to the local areas determined at the first step of the optimization.

\[
\text{Cost}([T]) = \sum_{i=1}^{N_{\text{data}}} ([T] \bar{S}(u_i, v_i) - \bar{P}_i) \cdot ([T] \bar{S}(u_i, v_i) - \bar{P}_i ) \tag{3.4}
\]

The optimization iterates by finding the closest mesh points, minimizing the closest point distances and calculating a Newton step in the alignment variables. The process continues until a maximum number of iterations is reached, the maximum data error falls within the fit tolerance or the difference between cost function values at each step falls below a given value.

3.3 Feature-Based Deformation

The primary goal of the feature-based fitting process is to fit the data to within a tolerance while migrating the inherent features in the model to the corresponding features of the data. The optimized alignment, scaling and closest point partitioning routines mentioned above provide a rough correlation between the model and data. From this initial sizing, the model is deformed iteratively using the feature points selected during the model construction. This feature-based deformation keeps the relational position of the surface patches and places the model vertices in a suitable position for subsequent optimization.

To start the process, data points are associated with each feature point defined on the surface. During model creation, a parametric neighborhood or
region of influence is defined for each feature. Any closest point falling within a feature point's domain becomes associated with that feature point. For example, Figure 3.2 shows the associated closest points for a nose feature on a facial profile model. In a manner similar to the method proposed by Marilyn Lord [LORD87], the surface is iteratively deformed using the average of the error vectors within each parametric range. The averaged vectors, as shown in Figure 3.3, indicate the approximate distance that respective feature points must be moved to better fit the data.

![Nose range markers](image)

**Figure 3.2** Picture of the nose feature with parametric range and associated closest point error vectors

Since the feature points are functions of the control vertices, a change in the feature points must correlate to a change in the control vertex positions. If the weights of the vertices and the parametric coordinates of the feature points are held constant, the following linear system defines the change in the feature point positions, $\Delta F$, as a function of vertex deformations, $\Delta V$, based on the constant matrix of basis function values, $[B]$.

$$[B][\Delta V] = \{\Delta F\}$$  \[3.5\]
In most cases, the number of feature points is much less than the number of control vertices defining the model. Therefore, the feature changes represent an under-constrained system with \( m \) equations and \( n \) unknowns. Applying a Singular Value Decomposition (SVD) technique, \( n-m \) independent solutions are possible [SCHW89, WATK91]. Since the objective of the feature-based deformation is to maintain the structure of the model during the fitting process, the solution that changes the vertices the least is applied [PRES95].

The closest point parameters, the averaged feature point deflection vectors and the associated vertex deformations are recalculated at each iteration of the deformation procedure. The entire process is repeated until the sum of the squared data point distances converges or a limiting number of iterations is reached.
3.4 Vertex Optimization

The result of the feature point deformation algorithm is a model that roughly approximates the data. Due to the nature of the SVD calculations, the original structure of the model remains intact. With the model in this state, a vertex optimization is applied to fit the model to within the desired tolerance. The least squares cost function, given in Equation 3.6, is a function of the vertex positions and weights and the closest point parameters.

\[
\text{Cost}(\vec{V}, w, u, v) = \sum_{i=1}^{\text{N\text{data}}} \left( \vec{s}(\vec{V}, w, u_i, v_i) - \vec{P}_i \right) \cdot \left( \vec{s}(\vec{V}, w, u_i, v_i) - \vec{P}_i \right)
\]  

[3.6]

A least squares formulation is analytic and smoothes the error within a data set. The norm assumes that no large outliers are present in the data, a practical assumption if the source of the data is laser scans.

Because of the large number of variables that are optimized, a Newton-type method is very computationally expensive. This is a result of having to calculate the Hessian matrices and to solve a large linear system at each iteration. Therefore, a comparable, Polak-Ribiere conjugate gradient optimization is implemented, as suggested by Laurent [LAUR93]. To overcome ill-conditioning of the system, Laurent also recommends that the vertex positions be solved separately using a linear least squares technique. However, a large linear system solution dramatically increases the computation time of the algorithm. For this reason, a small conditioning problem is accepted to achieve a more efficient algorithm.

Four versions of the optimization are implemented to determine which procedure provides the best solution to tolerance. The first optimizes the vertex positions, weights and closest point parameters simultaneously. The second version calculates the closest point parameters separately at each iteration of the
conjugate gradient optimization. The remaining two versions are the same as the first two except that the weights are not included in the optimization. For the two procedures which include the vertex weights, the penalty shown in Equation 3.7 is added to the cost function keep the weights positive and within a limited range [DIXO72, GILL89]. The optimization continues until the maximum error falls within tolerance, the function converges, or a maximum number of iterations is reached.

\[
\text{Penalty} = \sigma \sum_{i=1}^{N_{\text{vertices}}} \left( a_i(w_i) f_i(w_i) + b_i(w_i) g_i(w_i) \right) \quad [3.7]
\]

where

\[
\begin{align*}
\sigma &= 10,000; \text{ penalty scale value} \\
f_i(w_i) &= w_{\text{min}} - w_i \\
g_i(w_i) &= w_i - w_{\text{max}} \\
a_i(w_i) &= \begin{cases} 0, & f_i(w_i) \leq 0 \\ 1, & f_i(w_i) > 0 \end{cases} \\
b_i(w_i) &= \begin{cases} 0, & g_i(w_i) \leq 0 \\ 1, & g_i(w_i) > 0 \end{cases}
\end{align*}
\]

The optimization routine presented above is sensitive to the starting location. The preliminary rough fit of the data ensures that the local minimum found by this minimization maintains the structure contained in the original model. Optimizing the model without this initial deformation can produce substantially different and unsatisfactory results.

### 3.5 Hierarchical Refinement

If the vertex optimization does not fit the model to within the desired tolerance, hierarchical refinement is used to locally adjust the surface patches that contain data which are out of tolerance [FORS88]. This refinement minimizes the number of vertex additions necessary for each localized patch deformation without affecting the continuity with neighboring patches.
For example, the central span of the 2-D curve shown in Figure 3.4 is to be subdivided into four smaller spans. The parameter range defining the span is redefined with three new knots added. The neighboring curve spans, however, continue to use the original, unedited knot vector. New control vertices, shown as open circles in Figure 3.5, are calculated using the OSLO algorithm such that the refined set of spans is equivalent to the original [BART87]. To preserve the continuity between the neighboring spans, only the central, newly created vertex is allowed to move, as shown in Figure 3.6. If two neighboring spans are refined in this manner, then the three new vertices along the common boundary must remain coincident with the respective three in the next span to ensure continuity. As a result, the two central vertices as well as the three common vertices in between may be repositioned without damaging continuity.

Figure 3.4  B-spline curve span (bolded) to be subdivided using hierarchical refinement.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 3.5  Hierarchical refinement of central span. New vertices are shown as large open circles.

Figure 3.6  Deformed hierarchical curve span. The original curve is shown as a dashed line.

After the vertex optimization, any patch containing a data point that is out of tolerance is subdivided and the moveable vertex positions and weights optimized. This refinement and optimization sequence is repeated until all of the surface patches are within tolerance or a set number of hierarchical levels is applied.

3.6  Sample Applications and Results

To illustrate the concepts of the algorithm presented, the feature-based fitting procedure was initially implemented using two-dimensional curves. This
laid the groundwork for the subsequent extension into three-dimensional surface fitting [DOBS95]. Both applications are discussed in the following sections.

3.6.1 Pilot Study (2-D Facial Profiles)

As a preliminary study, a feature-based model was applied to a set of facial silhouette profiles. Data points were collected by digitizing hand-traced shadows of human profiles. On average, each set consisted of 73 sequential data points. The generic model was constructed by a linear, least-squares fit of a uniform cubic rational B-spline to one of the data sets (Profile 5). Spans on the spline curve were forced to correspond to specific areas of the face (i.e. the nose, chin, brow, etc.). The resulting model, shown in Figure 3.7, consists of 16 vertices, with all weights set to unity. Although not required in this study, the technique works equally well on models incorporating multiple knots, should endpoint interpolation or continuity reduction be required.

The feature points, shown in Figure 3.7 as heavy, dark dots, are an adaptation of those used by Harmon and Hunt [HARM77, HARM78] for automatic facial feature extraction. From the top of the profile down, the fiducial points are:

1. brow (maximum extension above the eye),
2. bridge of nose,
3. tip of the nose,
4. base of the nose (point of 45° tangency below the tip of nose),
5. protrusion of the upper lip,
6. cusp of the mouth,
7. protrusion of the lower lip, and
8. maximum protrusion of the chin.
Figure 3.7  Facial profile model comprised of 16 control vertices. The vertices are offset horizontally for clarity.
The associated feature measures are defined as the following distances between the feature points on the model:

1. brow point to tip of the nose,
2. bridge of the nose to the tip of the nose,
3. tip of the nose to the base of the nose,
4. tip of the nose to the cusp of the mouth,
5. tip of the nose to the upper lip point,
6. upper lip point to lower lip point,
7. tip of the nose to the chin point.

Harmon and Hunt discuss a number of other features such as baseline, profile area, and base angle of the profile triangle, which are directly obtainable from the list of model feature points.

Since the cusp of the mouth is an easily recognized and flagged feature, the model is positioned with the data using this landmark. The mouth cusp point is the sole landmark taken from each data set. The initial orientation and sizing of the model to the data is achieved by aligning the principal axes of the model to those of the data and scaling by the minimum and maximum extensions along the primary axes of the model. After scaling, a univariate least squares optimization calculates the angle of rotation about the cusp producing the best alignment based on the metric given by Equation 3.4. Feature-based deformation and vertex position and weight optimization proceed as discussed previously.

The intermediate fits produced after the completion of the alignment and feature deformation stages of the process are shown for two data sets in Figures 3.8 and 3.9. The maximum distance from the model to each data set and the CPU time on a VAX 7600 for each stage are presented in Table 3.1. To date,
the two-dimensional feature-based fitting procedure has not been optimized for computational efficiency. The final tolerance fits, control polygons, and vertex weights for a collection of facial profiles are shown in Figures 3.10 - 3.14.

Table 3.1 Maximum distances (inches) from the model to data and CPU times (sec) on a VAX 7600 for each stage of the feature-based modeling process.

<table>
<thead>
<tr>
<th>Profile</th>
<th>Alignment</th>
<th>Feature Fit</th>
<th>Tolerance Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dist</td>
<td>time</td>
<td>dist</td>
</tr>
<tr>
<td>1</td>
<td>1.17</td>
<td>3.92</td>
<td>0.27</td>
</tr>
<tr>
<td>2</td>
<td>0.36</td>
<td>4.05</td>
<td>0.21</td>
</tr>
<tr>
<td>3</td>
<td>0.64</td>
<td>3.80</td>
<td>0.24</td>
</tr>
<tr>
<td>4</td>
<td>0.24</td>
<td>4.13</td>
<td>0.23</td>
</tr>
<tr>
<td>5</td>
<td>0.27</td>
<td>4.22</td>
<td>0.09</td>
</tr>
</tbody>
</table>

Figure 3.8 Alignment and feature fitting stages prior to tolerance fit for Profile 1: (a) Alignment and initial sizing; (b) Feature point deformation.
Figure 3.9 Alignment and feature fitting stages prior to tolerance fit for Profile 2: (a) Alignment and initial sizing; (b) Feature point deformation.
Figure 3.10 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 1.
Figure 3.11  Final tolerance fit with horizontally offset control vertices and associated weights for Profile 2.
Figure 3.12 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 3.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 3.13 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 4.
Figure 3.14 Final tolerance fit with horizontally offset control vertices and associated weights for Profile 5.
The quality of the initial fit produced by alignment and scaling varies significantly from subject to subject due to the varying shape of the individual profiles and the extent to which the forehead and underside of the chin are digitized. In each case, however, the initial fit is sufficient to produce an acceptable partition for the first iteration of the feature deformation. During subsequent steps, appropriate regions of the model migrate toward their counterparts in the data. The final feature-based fits result in close approximations to the data while keeping the spans in their proper correspondence (i.e. the model's span at the tip of the nose stays on the tip of the nose of the data). Vertex optimization further improves the fit without altering the relationships between the feature areas.

The vertex weights are constrained between 0.1 and 5.0; however, all vertex weights remain within this range without the penalty being invoked. The tolerance for the final fit is set to 0.04 inches, which is smaller than the resolution of the data acquisition. This is done to illustrate the hierarchical refinement stage which, otherwise, would not have been invoked for these particular data sets (jowl of Profiles 2 and 5; brow of Profile 3). The final feature values, as defined earlier, are shown in Table 3.2 for each final fit.

Table 3.2 Feature point distances for the sample profiles in Figures 3.10-3.14. All measures are in inches.

<table>
<thead>
<tr>
<th>Profile Measure</th>
<th>Profile 1</th>
<th>Profile 2</th>
<th>Profile 3</th>
<th>Profile 4</th>
<th>Profile 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brow to Nose Tip</td>
<td>2.54</td>
<td>2.73</td>
<td>1.98</td>
<td>2.83</td>
<td>2.69</td>
</tr>
<tr>
<td>Bridge to Nose Tip</td>
<td>2.17</td>
<td>2.43</td>
<td>1.59</td>
<td>2.16</td>
<td>2.24</td>
</tr>
<tr>
<td>Tip to Base of Nose</td>
<td>0.62</td>
<td>0.87</td>
<td>0.72</td>
<td>0.92</td>
<td>0.84</td>
</tr>
<tr>
<td>Nose Tip to Mouth Cusp</td>
<td>1.30</td>
<td>1.86</td>
<td>1.60</td>
<td>2.02</td>
<td>1.88</td>
</tr>
<tr>
<td>Nose Tip to Upper Lip</td>
<td>0.83</td>
<td>1.48</td>
<td>1.04</td>
<td>1.57</td>
<td>1.49</td>
</tr>
<tr>
<td>Upper to Lower Lip</td>
<td>0.74</td>
<td>0.58</td>
<td>0.80</td>
<td>0.66</td>
<td>0.65</td>
</tr>
<tr>
<td>Nose Tip to Chin</td>
<td>2.99</td>
<td>3.33</td>
<td>3.11</td>
<td>3.22</td>
<td>3.44</td>
</tr>
</tbody>
</table>
3.6.2 Three-Dimensional Surface Fitting

The pilot study, using two-dimensional models, shows that the concepts of the feature-based fitting procedure have merit. Further development extrapolates the algorithm to the fitting of three-dimensional surfaces. As a sample implementation, a model of the human foot is used to fit data sets acquired using laser scanners and to extract measures commonly used in the fitting of shoes.

An initial model is constructed using the data shown in Figure 2.1, a 5,100 point cloud set provided by Cyberware scanning company. To simplify the parameterization of the data, slices are created to represent iso-parametric contours, shown in Figure 3.15. A single parametric value is specified for each slice representing one parametric direction. Within each slice, the data points are parameterized using the centripetal algorithm presented by Lee to constitute the second parametric coordinate [LEE89]. The resulting surface from a linear least squares solution is shown in Figure 3.16. Defined by an 18 x 25 grid of vertices and only one knot vector in each parametric direction, the NURB model is represented by a single collection of surface patches folded upon itself at the sides of the foot. The vertices along the seam are constrained to enforce curvature continuity along the adjoining patches.

Parametric points on the surface are chosen to represent characteristic features of the foot. These feature points are shown in Figure 3.17. Also shown are the parametric ranges bounding each feature point for use in the feature-point deformation routine. Using the defined feature placements, measurements common to the footwear industry are easily imposed and extracted, as seen in Figures 3.18-3.19 [ROSS93]. Currently, there are no standard, foot measurements defined within the shoe industry.
Figure 3.15  Iso-parametric slices from the data set given in Figure 2.1.

Figure 3.16  Surface constructed using a least squares fit to the slices given in Figure 3.15.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 3.17  Defined feature points with associated parametric ranges.

Figure 3.18  Point-to-point measures and ball and waist girths. The model is displayed as a grid of patches for ease of viewing.
Figure 3.19  Projected measures for a foot. The model is displayed as a grid of patches for ease of viewing.

The initial alignment of the model to the data depends on the structure of the data set. If the set is a cloud of points with no order provided, then the center of mass is calculated as the average of the three-dimensional coordinate values. The moments of inertia are determined by assuming each point is a point mass and applying the method presented by Patrick Lord [LORD94]. Although this technique is simple and computationally efficient, the resulting alignment depends directly on the distribution of the data points on the surface. The calculated moments will be biased to those areas that contain more points. In practice, this shortcoming will not be encountered frequently since laser and volume scanners provide evenly distributed and extremely dense data sets.

If the data set has a supplied structure, the volume is estimated by a series of tetrahedra (defined by a triangular surface facet and a reference point).
The center of mass is first calculated by summing the volumes and center of mass products. The moments are calculated relative to this center of mass by summing the moments of a second tetrahedron series constructed in reference to the center of mass. This method is not affected by point distribution. The only limiting factor is the extent to which the data accurately represent the underlying surface. The differing results from these two methods are discussed in Chapter 5. A similar technique is used to calculate the center of mass and moments for the model. The surface is discretized at a specified parametric step and then considered a structured data set.

A data set provided by a colleague who manually sliced and digitized a plaster cast of his foot is used to demonstrate the 3-D surface fitting process. The resulting slice data set and the constructed foot model are displayed in Figure 3.20. The data and model are shown in their respective defined orientations. The initial alignment collocates the model to the data by the centers of mass and aligns the respective principal axes. The initial center of mass and principal axes alignment for the data set and the model in Figure 3.20 is shown in Figure 3.21.

Given this initial orientation, the position, rotation and scale variables of the model are optimized to improve the alignment. Using a specified discretization of the surface, the closest mesh point on the surface is determined for each data point. The parametric value of each closest mesh point is used to seed a Newton optimization of the closest point on the surface to each data point. The resulting errors in the fit are combined into the least squares norm of Equation 3.4 for subsequent optimization. Since the rotation angles may change by as much as 45 degrees, the closest mesh point seeds are determined at each iteration of the optimization. Also, practice shows that the optimization
Figure 3.20 Foot model and slice data set positioned as defined.

Figure 3.21 Initial alignment of the foot model to the data using center of mass and moment calculations only.
of the scale parameters should be postponed until the adjustments in the rotation angles have been minimized (less than one angular degree of rotation). This insures that a Newton step in the scale does not push the solution into an unwanted minimum. Figure 3.22 shows the optimized alignment of the model to a data set starting from the initial alignment shown in Figure 3.21.

![Optimized alignment for the model and data set shown in Figure 3.21.](image)

The next step in the feature-based fitting algorithm deforms the model using the features inherent to its description. The closest points on the surface are determined for each data point. These closest point parameters are checked against the effective ranges of the feature points to associate the data to a specific feature. For each feature, the associated error vectors from the closest points to the data points are averaged. The calculated averages are used in
Equation 3.5 to solve for the necessary changes in the control vertices. The process is repeated until the sum of the squared errors converges or a maximum number of iterations is reached. The resulting deformation for the fit displayed in Figure 3.22 is shown in Figure 3.23.

Figure 3.23  Feature point deformation for the model and data set shown in Figure 3.22.

Subsequent vertex position and weight optimization improves the fit without altering the relationships between the features on the model and data set, as seen in Figure 3.24. Weights were limited between 0.1 and 5.0; however, all vertex weights remained within this range without the penalty being invoked. The tolerance for the final fit was set to 0.08 inches, a value that is three times smaller than the increments used in shoe fitting. Once the data is fit to the tolerance specified, the desired measures, shown graphically in Figures 3.25 and 3.26, are easily extracted using the associated features.
Figure 3.24  Final tolerance fit for the model and data set shown in Figure 3.23.

Figure 3.25  Point-to-point distances and girth (circumference) measures for the final fit shown in Figure 3.24.
The final tolerance fits for three additional foot scans are shown in Figures 3.27 – 3.29, respectively. The data sets (point clouds) were provided by two private scanning companies. True measures are available only for the last two sets (Figures 3.28 and 3.29). Comparisons to the extracted values are deferred until Chapter 4. Table 3.3 provides the associated maximum errors and computation times on a Silicon Graphics O2 machine for each subprocess of the algorithm. The final feature values for the four data sets are shown in Table 3.4. The maximum number of iterations for the alignment, feature point and final tolerance optimizations are 60, 20, and 100, respectively.

The four data sets have 4088, 4215, 4396, and 4350 data points, respectively. The last three data sets represent socked feet. Therefore, a second model, without toe definition, is fit to those data sets. The new model is defined
Figure 3.27 Final tolerance fit for Foot 2.

Figure 3.28 Final tolerance fit for Foot 3.
Figure 3.29  Final tolerance fit for Foot 4.

Table 3.3  Maximum errors (inches), number of iterations and run times (sec) on a Silicon Graphics O2 machine (180 MHz) for the four feet shown in Figure 3.24, and Figures 3.27-3.29 (Feet 1-4, respectively).

<table>
<thead>
<tr>
<th>Foot</th>
<th>#Pts.</th>
<th>Alignment iter.</th>
<th>time</th>
<th>error</th>
<th>Feature Fit iter.</th>
<th>time</th>
<th>error</th>
<th>Tolerance Fit iter.</th>
<th>time</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4,088</td>
<td>31</td>
<td>21</td>
<td>0.32</td>
<td>2</td>
<td>17</td>
<td>0.35</td>
<td>94</td>
<td>270</td>
<td>0.07</td>
</tr>
<tr>
<td>2</td>
<td>4,215</td>
<td>60</td>
<td>43</td>
<td>0.51</td>
<td>4</td>
<td>11</td>
<td>0.56</td>
<td>100</td>
<td>290</td>
<td>0.11</td>
</tr>
<tr>
<td>3</td>
<td>4,396</td>
<td>60</td>
<td>39</td>
<td>0.49</td>
<td>20</td>
<td>30</td>
<td>0.44</td>
<td>100</td>
<td>274</td>
<td>0.10</td>
</tr>
<tr>
<td>4</td>
<td>4,350</td>
<td>60</td>
<td>50</td>
<td>0.38</td>
<td>20</td>
<td>32</td>
<td>0.57</td>
<td>100</td>
<td>275</td>
<td>0.10</td>
</tr>
</tbody>
</table>
Table 3.4  Extracted measures for the data in Figures 3.24 and 3.27-3.29, respectively. All values are in inches.

<table>
<thead>
<tr>
<th>Point Distances (in.)</th>
<th>Foot 1</th>
<th>Foot 2</th>
<th>Foot 3</th>
<th>Foot 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heel to First Toe</td>
<td>11.24</td>
<td>10.22</td>
<td>11.33</td>
<td>11.42</td>
</tr>
<tr>
<td>Heel to Inner Ball</td>
<td>8.38</td>
<td>7.56</td>
<td>8.06</td>
<td>8.35</td>
</tr>
<tr>
<td>Heel to Outer Ball</td>
<td>7.89</td>
<td>6.54</td>
<td>7.04</td>
<td>7.28</td>
</tr>
<tr>
<td>Inner Ball to Outer Ball</td>
<td>4.10</td>
<td>3.99</td>
<td>4.30</td>
<td>4.39</td>
</tr>
<tr>
<td>Inner to Outer Heel</td>
<td>2.47</td>
<td>2.42</td>
<td>2.78</td>
<td>2.82</td>
</tr>
<tr>
<td>Inner-to-Outer Ankle</td>
<td>2.88</td>
<td>3.12</td>
<td>3.20</td>
<td>3.33</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Projected Measures (in.)</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>11.15</td>
<td>10.11</td>
<td>11.26</td>
</tr>
<tr>
<td>Inner Ball Length</td>
<td>8.02</td>
<td>7.26</td>
<td>7.75</td>
</tr>
<tr>
<td>Outer Ball Length</td>
<td>7.70</td>
<td>6.25</td>
<td>6.73</td>
</tr>
<tr>
<td>Heel Length</td>
<td>1.88</td>
<td>1.23</td>
<td>1.55</td>
</tr>
<tr>
<td>Ball Width</td>
<td>4.08</td>
<td>3.84</td>
<td>4.16</td>
</tr>
<tr>
<td>Heel Width</td>
<td>2.40</td>
<td>2.41</td>
<td>2.78</td>
</tr>
<tr>
<td>Ankle Width</td>
<td>2.84</td>
<td>3.06</td>
<td>3.16</td>
</tr>
<tr>
<td>Inner Arch Height</td>
<td>1.39</td>
<td>1.69</td>
<td>1.52</td>
</tr>
<tr>
<td>Outer Arch Height</td>
<td>0.70</td>
<td>0.77</td>
<td>0.77</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Circumferences (in.)</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ball Girth</td>
<td>9.99</td>
<td>10.01</td>
<td>10.61</td>
</tr>
<tr>
<td>Instep Girth</td>
<td>10.55</td>
<td>10.90</td>
<td>11.26</td>
</tr>
</tbody>
</table>

defined by a 15 x 27 grid of vertices (45 less vertices than the "toed" model used on the first data set). The final tolerance optimization was run only to a maximum of 100 iterations. This prohibited the last three models from reaching the desired data tolerance (0.08 inches). Subsequent testing showed that the tolerance was met after approximately 130 iterations for each data set.

Feet 3 and 4 (Figures 3.28 and 3.29) show a large undulation at the rear of the foot. This is a side effect of the final tolerance optimization. Since no points are present at the top-rear of either data set, the vertices defining those patches of the model are allowed to float. No constraint is placed on patches...
that have no data points associated with them. Therefore, these "free" vertices may be displaced adversely as a consequence of fitting points on neighboring surface patches. A similar effect appears in the final fit of Foot 2 (Figure 3.27). The data set does not contain points as far up the leg as the model is defined. The deformations apparent at the top of the foot are a result of fitting the data points below the ankle.

Hierarchical refinement is not incorporated into the three-dimensional surface fitting algorithm. Had the refinement technique been included, patches that were out of tolerance on the last three models would have been subdivided to increase the flexibility of fit. The newly created vertices are then sent through a similar optimization to improve the final fit.

3.7 Summary

Each component of the feature-based fitting algorithm is outlined in this chapter. A least squares fit to a sample data set constructs the base or "generic" model surface. Characteristic features are selected and embedded within the model. Once constructed, the model is then fit to a supplied data set by first aligning the model to the data, matching features and optimizing the vertex positions and weights.

As shown in the chapter, the algorithm works for the two applications presented (2-D profiles and 3-D foot models). However, no proof is given for accuracy and repeatability of the measures. These topics are addressed in the next chapter where each component of the algorithm is tested on data sets of various sizes (number of data points) and structure (cloud vs. slices).
Chapter 4 Algorithm Testing

The previous chapter shows that the feature-based modeling algorithm is able to fit a surface to a data set and extract information. However, to this point, no proof has been given to show that the recorded measures are accurate or reproducible. This section tests each component of the feature-based fitting algorithm for sensitivities and limitations and compares the extracted measures for accuracy and repeatability.

As stated previously, the structure of the data set (cloud/slice) determines the method for calculating the principal moments. Section 4.1 demonstrates the differences in initial alignment parameters by treating two sets (Figure 3.20, and a version with fewer slices) as both cloud and slice data. Repeatability tests are also run to test the sensitivity of the optimized alignment to data density and distribution. The six data sets used in this study were provided by Laser Design and Red Wing Shoe Company. The data sets are derived from a single foot scan by eliminating points from the original scan using a marching box tolerance applied by the scanning company. Details about the actual tolerances used by the scanning company are currently unknown to the author.

The second section discusses the limitations of the feature-point deformation procedure. Potential corrective measures are also presented. Section 4.3 shows how the errors in the alignment and feature-deformation procedures cascade into the vertex optimization. Problems resulting from missing areas of data and open-end models are also discussed. Four versions of the greatest conjugate gradient vertex optimization are implemented. The computation times, cost functions and maximum errors of each option are compared for the six data sets presented in the optimized alignment tests.
These same six data sets are used to test the repeatability of the measure extraction portion of the algorithm (Section 4.4). A seventh data set (a rotated version of the second set) is included in the repeatability tests. Unfortunately, no exact measures are known for these data sets. The accuracy of the extracted measures is tested at the end of the section using the two data sets shown in Figures 3.28 and 3.29, whose actual dimensions were supplied by Brightwood Shoe Company. The chapter ends with a discussion of the limitations encountered and presents appropriate explanations.

4.1 Registration

Excluding the use of landmarks, the choice of initial alignment depends on the structure of the supplied data set. If the data set is provided as a series of slices, then the principal moments are calculated using the tetrahedra summation technique provided by Lien [LIEN84]. Otherwise, the data set is considered as a random cloud of points and the moments are determined by the point-mass technique presented by Patrick Lord [LORD94]. If the data set accurately represents the acquired object, the volume summation results tend to be more accurate than the point-mass alignment, by as much as 20% in position and 10% in rotation. To test the differences in the two alignment strategies, the data set used in Figure 3.20 (Figure 4.1, 4088 points) is treated as both a slice and cloud data set. Additionally, a second version of the data set, with fewer slices (Figure 4.2, 1095 points), is also tested. The initial rotation angles and positions for the four data versions are compared to the optimized values, as seen in Table 4.1. The corresponding initial alignments for each test are provided in Figures 4.3-4.6.
Figure 4.1 Data set consisting of 4088 data points. The data set is used in the initial alignment testing of Section 4.1.
Figure 4.2  Data set consisting of 1,095 data points. The data set is used in the initial alignment testing of Section 4.1
Figure 4.3 Initial alignment of the data set consisting of 1095 points, taken as a point cloud.
Figure 4.4 Initial alignment of the data set consisting of 4088 points, taken as a point cloud.
Figure 4.5 Initial alignment of the data set consisting of 1095 points, taken as a series of slices.
Figure 4.6 Initial alignment of the data set consisting of 4088 points, taken as a series of slices.
Table 4.1  Initial position and rotation angles (degrees) for the four test data sets. The associated
differences and relative, percentage errors from the optimized values are also given. The
position percentages are scaled relative to an 11.2" foot length and the angular percentages to a
full 360 degree rotation.

<table>
<thead>
<tr>
<th>Opt. Position Value</th>
<th>Cloud 1095 Value</th>
<th>Diff.</th>
<th>% Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.40</td>
<td>0.09</td>
<td>0.8%</td>
</tr>
<tr>
<td>Y</td>
<td>-2.85</td>
<td>0.05</td>
<td>0.4%</td>
</tr>
<tr>
<td>Z</td>
<td>0.63</td>
<td>1.34</td>
<td>11.9%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rotation Value</th>
<th>Cloud 4088 Value</th>
<th>Diff.</th>
<th>% Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>-6.78</td>
<td>2.53</td>
<td>0.7%</td>
</tr>
<tr>
<td>Y</td>
<td>9.75</td>
<td>8.50</td>
<td>2.4%</td>
</tr>
<tr>
<td>Z</td>
<td>-17.14</td>
<td>24.58</td>
<td>6.8%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value Diff.</th>
<th>% Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Diff.</td>
</tr>
<tr>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>0.51</td>
<td>0.20</td>
</tr>
<tr>
<td>-2.94</td>
<td>0.15</td>
</tr>
<tr>
<td>1.50</td>
<td>2.21</td>
</tr>
<tr>
<td>-0.28</td>
<td>3.97</td>
</tr>
<tr>
<td>-7.88</td>
<td>9.14</td>
</tr>
<tr>
<td>51.22</td>
<td>43.78</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Opt. Position Value</th>
<th>Cloud 1095 Value</th>
<th>Diff.</th>
<th>% Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.31</td>
<td>0.00</td>
<td>0.0%</td>
</tr>
<tr>
<td>Y</td>
<td>-2.93</td>
<td>0.14</td>
<td>1.2%</td>
</tr>
<tr>
<td>Z</td>
<td>-0.61</td>
<td>0.03</td>
<td>0.2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rotation Value</th>
<th>Cloud 4088 Value</th>
<th>Diff.</th>
<th>% Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>-3.53</td>
<td>0.72</td>
<td>0.2%</td>
</tr>
<tr>
<td>Y</td>
<td>0.99</td>
<td>0.27</td>
<td>0.1%</td>
</tr>
<tr>
<td>Z</td>
<td>8.57</td>
<td>1.12</td>
<td>0.3%</td>
</tr>
</tbody>
</table>
Because the point-mass technique is sensitive to the distribution of the data, the initial alignment calculated is less accurate than the volume summation. Collecting more data points does not necessarily result in a better alignment. The cloud data set with 4088 data points has larger alignment errors than the cloud set with only 1095 points. The reason is that the larger data set has more densely packed points at the toes as compared to the rest of the foot surface. This biases the point-mass calculations toward the toes.

The volume approximation technique is not sensitive to point distribution. Its only requirement is that the slices accurately represent the contours of the surface. Both slice data sets show much smaller errors than the point-mass alignments. However, the set with fewer slices has slightly larger errors than the full slice set because it does not capture the curvature at the heel or the extent of the toes accurately.

Even with these variations, the initial alignment places the model in a position suitable for the subsequent optimization routine. This stage in the process is tested for repeatability using six cloud data sets representing the same foot in the same position. The data sets, provided by a joint venture between Laser Design, Inc. and Red Wing Shoe Company, were created from a single, dense data set by applying a marching box, deviation filter similar to the one presented by Odesanya [ODES93]. The filter creates a line segment from the first point on a slice to the third point. If the distance from the second point on the slice to the segment is less than the tolerance, the segment is redrawn from the first to the fourth point on the slice. The distances from the second and third points to the segment are tested again. The process continues until one of the interior points is out of tolerance. At which point, the end points from the previous, “good” segment are kept and the interior points discarded. The
process begins anew with the last end point and proceeds until the slice is traversed. Unfortunately, the actual tolerances used by the scanning company to create the six data sets are not disclosed. Images of the six data sets are provided in Appendix 1 as Figures A.1-A.6.

The optimized alignment parameters are given in Table 4.2 for each data set. The largest errors (deviations) in the alignment sensitivity tests occur in the rotation angles. The roll angle (z-axis, along the length of the foot) shows the largest percentage error at 0.7% with a deviation of 2.3 degrees. The x-axis position component shows an error of 0.5%, yet the deviation is only 0.06 inches. The reason for the contradiction is that the angular percentages are taken relative to a full 360 degree rotation. The position percentage errors are taken relative to the total foot length, 11.7 inches in this case. The scale difference is also demonstrated by the scale factors. The maximum deviation is 0.01 but the percentage errors (relative to 1.0) are nearly twice those of the position and rotation values. Since the foot model is roughly the same size as the data set, the global scales show little deviation from unity.

Because the alignment optimization only allows a global scale along the foot axes, the routine may not be able to adequately match the model to the data. For example, the foot represented by the six data sets in Table 4.2 has a much smaller ball width than the model; however, the other components of the foot are similar in size. Scaling by a global parameter alone prohibits the optimization from accurately matching the model to the inner ball of the data set, as shown in Figure 4.7. Separate component scales or a more intelligent optimization may be required to correct this problem. Compounding the problem, the laser scanner is not able to collect points at the tip of the large toe, which may allow the model to scale beyond the correct toe box.
Table 4.2 Optimized alignment parameters for six versions of the same foot scan. Alignment rotation angles are in degrees. Position percentages are relative to a foot length of 11.7 inches and the angle percentages are relative to a full 360 degree rotation.

<table>
<thead>
<tr>
<th>Set</th>
<th># Points</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3,109</td>
<td>-0.22</td>
<td>0.19</td>
<td>1.74</td>
<td>-91.1</td>
<td>93.3</td>
<td>181.2</td>
<td>1.05</td>
<td>0.99</td>
<td>1.08</td>
</tr>
<tr>
<td>2</td>
<td>4,713</td>
<td>-0.37</td>
<td>0.17</td>
<td>1.72</td>
<td>-91.6</td>
<td>94.2</td>
<td>182.7</td>
<td>1.04</td>
<td>0.99</td>
<td>1.08</td>
</tr>
<tr>
<td>3</td>
<td>17,091</td>
<td>-0.31</td>
<td>0.16</td>
<td>1.80</td>
<td>-95.4</td>
<td>94.5</td>
<td>186.2</td>
<td>1.02</td>
<td>1.02</td>
<td>1.06</td>
</tr>
<tr>
<td>4</td>
<td>2,353</td>
<td>-0.36</td>
<td>0.19</td>
<td>1.75</td>
<td>-91.7</td>
<td>94.0</td>
<td>180.6</td>
<td>1.05</td>
<td>1.00</td>
<td>1.09</td>
</tr>
<tr>
<td>5</td>
<td>9,965</td>
<td>-0.27</td>
<td>0.20</td>
<td>1.74</td>
<td>-92.7</td>
<td>92.3</td>
<td>182.0</td>
<td>1.04</td>
<td>1.01</td>
<td>1.10</td>
</tr>
<tr>
<td>6</td>
<td>5,179</td>
<td>-0.30</td>
<td>0.19</td>
<td>1.75</td>
<td>-90.2</td>
<td>92.6</td>
<td>179.4</td>
<td>1.05</td>
<td>1.02</td>
<td>1.08</td>
</tr>
</tbody>
</table>

Mean

<table>
<thead>
<tr>
<th>Position</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-0.30</td>
<td>0.18</td>
<td>1.75</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Angles (deg.)</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-92.1</td>
<td>93.5</td>
<td>182.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.04</td>
<td>1.01</td>
<td>1.08</td>
</tr>
</tbody>
</table>

Std. Dev.

<table>
<thead>
<tr>
<th>Position</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.06</td>
<td>0.01</td>
<td>0.03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Angles (deg.)</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.81</td>
<td>0.89</td>
<td>2.35</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
</tbody>
</table>

% Error

<table>
<thead>
<tr>
<th>Position</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.5%</td>
<td>0.1%</td>
<td>0.2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Angles (deg.)</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.5%</td>
<td>0.2%</td>
<td>0.7%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.1%</td>
<td>1.2%</td>
<td>1.0%</td>
</tr>
</tbody>
</table>
4.2 Feature Deformation

Once aligned, the next step in the feature-based fitting algorithm is the point deformation of the features. The routine averages the error vectors of the points with closest point parameters falling within the specified range of a feature. The resulting, averaged vectors determine how the feature points must move to better fit the data. These movements are translated into associated control vertex movements through Equation 3.5. To test the limitations of this procedure, the oriented models resulting from the six alignment optimization tests are deformed using the feature average vectors. The problems encountered are discussed below.

If no data points fall within the parametric range of a feature, that feature does not move. After the optimized alignment shown in Figure 4.7, the tip and side of the large toe features do not have data associated with them. Consequently, the feature-point deformation routine cannot correct the inaccurately enlarged toe box in the model. This error cascades into an
undesirable fit as will be discussed later. Implementing the component-specific scaling mentioned previously may correct this problem. Alternatively, the feature point deformations could be reworked into another optimization routine to minimize the total squared error in the data set.

Usually, the local manipulation property of B-splines is an advantage in data fitting. However, the point movements of the feature deformation routines may be too localized, resulting in peaks, valleys or folds in the surface. For example, the feature point deformation of the model in Figure 4.7 shows a bulge at the inner arch point and a fold at the top of the heel (see Figure 4.8). The desired effect is an expansion of the arch to the inside and a shift of the Achilles heel to the outside. Embedding the model within a free-form deformation mesh may broaden the effect of the feature point movements.

![Figure 4.8 Feature point deformation of the model in Figure 4.1. The image shows a localized bulge at the inner arch and a fold at the top of the heel.](image-url)
4.3 Tolerance Fitting

As stated earlier, both the optimized alignment and the feature point deformation routine failed to shift the inner side of the toe box to match the data set shown in Figure 4.7. Because the closest point optimization simply finds the point on the surface closest to each data point, the data points on the side of the foot in that region became associated with the top and bottom portions of the model. Therefore, the final vertex optimization did not know to shift the toe box to the inside. Consequently, the surface shrinks at the location of the points and pinches the model during the final tolerance optimization, as shown in Figure 4.9. This problem may be corrected by one of the previously mentioned suggestions in either the alignment or feature point routines or by adding more intelligence to the closest point optimization.

![Figure 4.9 Pinching of the model surface due to an initial misfit at the inside of the toe box.](image)
Logic dictates that the inclusion of weights within the model definition would allow greater flexibility in fitting a data set. This idea is tested using the current three-dimensional implementation of the feature-based modeling algorithm. A Polak-Ribiere greatest conjugate gradient (GCG) optimization routine is used to minimize the squared errors between the model and data set. Four variations of the GCG algorithm are tested for computation time and convergence properties:

1. Optimize vertex positions and weights, and closest point parameters within a GCG routine;

2. Optimize the vertex positions and weights parameters within a GCG routine, recalculate the closest point parameters after each GCG step using a Newton routine;

3. Optimize vertex positions and closest point parameters within a GCG routine, do not include weight optimization;

4. Optimize the vertex positions within a GCG routine, recalculate the closest point parameters after each GCG step using a Newton routine, do not include weight optimization.

To compare the four options, each algorithm is forced to take 100 steps while fitting the six test data shown in Figures A.1-A.6 in the Appendix. The computation times, sum of the squared errors and maximum errors for each trial are given in Table 4.3. As expected, the inclusion of the vertex weights provided additional flexibility to fit the data, as shown by the smaller maximum errors. In addition, optimization of the vertex weights has little effect on the computation time. Excluding the weights from the optimization results in a linear system of vertex positions; however, this linear solution is subject to singularities if there are large gaps in the data set, as seen in Figure 4.7. The
first option (all parameters within a GCG optimization) is the only procedure to fit all six data sets to within the 0.08-inch tolerance used in the previous fits.

Table 4.3 Run times (sec.), squared errors, and maximum error (inches) for the six data sets in Table 4.2 using the four different variations of the vertex optimization routine.

<table>
<thead>
<tr>
<th>NUMBER OF POINTS</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>3,109</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4,713</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17,091</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2,353</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9,965</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,179</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WEIGHTS</th>
<th>time</th>
<th>204</th>
<th>285</th>
<th>934</th>
<th>164</th>
<th>564</th>
<th>315</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCG only</td>
<td>sq. err.</td>
<td>0.30</td>
<td>0.76</td>
<td>2.05</td>
<td>0.38</td>
<td>1.45</td>
<td>0.72</td>
</tr>
<tr>
<td>max. err.</td>
<td></td>
<td>0.061</td>
<td>0.055</td>
<td>0.067</td>
<td>0.055</td>
<td>0.073</td>
<td>0.062</td>
</tr>
<tr>
<td>WEIGTHS</td>
<td>time</td>
<td>204</td>
<td>294</td>
<td>930</td>
<td>161</td>
<td>564</td>
<td>315</td>
</tr>
<tr>
<td>GCG only</td>
<td>sq. err.</td>
<td>0.29</td>
<td>0.75</td>
<td>2.31</td>
<td>0.43</td>
<td>1.42</td>
<td>0.73</td>
</tr>
<tr>
<td>max. err.</td>
<td></td>
<td>0.057</td>
<td>0.081</td>
<td>0.178</td>
<td>0.076</td>
<td>0.073</td>
<td>0.064</td>
</tr>
<tr>
<td>NO WEIGHTS</td>
<td>time</td>
<td>204</td>
<td>294</td>
<td>930</td>
<td>161</td>
<td>564</td>
<td>315</td>
</tr>
<tr>
<td>GCG only</td>
<td>sq. err.</td>
<td>0.38</td>
<td>0.91</td>
<td>2.05</td>
<td>0.43</td>
<td>1.55</td>
<td>0.83</td>
</tr>
<tr>
<td>max. err.</td>
<td></td>
<td>0.067</td>
<td>0.102</td>
<td>0.109</td>
<td>0.112</td>
<td>0.104</td>
<td>0.091</td>
</tr>
<tr>
<td>NO WEIGHTS</td>
<td>time</td>
<td>204</td>
<td>294</td>
<td>930</td>
<td>161</td>
<td>564</td>
<td>315</td>
</tr>
<tr>
<td>GCG/Newton</td>
<td>sq. err.</td>
<td>0.38</td>
<td>0.93</td>
<td>2.03</td>
<td>0.43</td>
<td>1.56</td>
<td>0.84</td>
</tr>
<tr>
<td>max. err.</td>
<td></td>
<td>0.07</td>
<td>0.14</td>
<td>0.10</td>
<td>0.09</td>
<td>0.10</td>
<td>0.09</td>
</tr>
</tbody>
</table>

4.4 Repeatability and Accuracy Tests

The same six data sets used in the previous section are used to test the repeatability of the extracted measures. Provided by an external supplier, the data sets represent the same foot scan, but vary in number of points. A seventh data set is added which is set number two rotated from its initial orientation (rotated by 110° x, -146° y and 12° z). The results of the feature-based fitting algorithm are given in Table 4.4.

The largest standard deviations occur in the heel-to-toe and overall lengths, 0.11 and 0.09 respectively. Since both measures are determined from
Table 4.4  Results of repeatability test for the data sets given the Appendix. All measures are in inches.

<table>
<thead>
<tr>
<th></th>
<th>Set 1</th>
<th>Set 2</th>
<th>Set 3</th>
<th>Set 4</th>
<th>Set 5</th>
<th>Set 6</th>
<th>Set 7</th>
<th>Mean</th>
<th>Std Dev</th>
<th>% Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td>3,109</td>
<td>4,713</td>
<td>17,091</td>
<td>2,353</td>
<td>9,965</td>
<td>5,179</td>
<td>4,713</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Point-To-Point</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Back of Heel to Toe</td>
<td>11.82</td>
<td>11.57</td>
<td>11.86</td>
<td>11.79</td>
<td>11.80</td>
<td>11.73</td>
<td>11.60</td>
<td>11.74</td>
<td>0.11</td>
<td>1.0</td>
</tr>
<tr>
<td>Heel to Inner Ball</td>
<td>8.64</td>
<td>8.46</td>
<td>8.62</td>
<td>8.62</td>
<td>8.62</td>
<td>8.60</td>
<td>8.47</td>
<td>8.58</td>
<td>0.08</td>
<td>0.9</td>
</tr>
<tr>
<td>Heel to Outer Ball</td>
<td>7.59</td>
<td>7.44</td>
<td>7.61</td>
<td>7.56</td>
<td>7.60</td>
<td>7.56</td>
<td>7.44</td>
<td>7.54</td>
<td>0.07</td>
<td>1.0</td>
</tr>
<tr>
<td>Inner to Outer Ball</td>
<td>4.33</td>
<td>4.39</td>
<td>4.31</td>
<td>4.36</td>
<td>4.35</td>
<td>4.36</td>
<td>4.35</td>
<td>4.35</td>
<td>0.03</td>
<td>0.6</td>
</tr>
<tr>
<td>Inner to Outer Heel</td>
<td>2.52</td>
<td>2.51</td>
<td>2.49</td>
<td>2.51</td>
<td>2.49</td>
<td>2.50</td>
<td>2.49</td>
<td>2.50</td>
<td>0.01</td>
<td>0.5</td>
</tr>
<tr>
<td>Inner to Outer Ankle</td>
<td>3.17</td>
<td>3.16</td>
<td>3.16</td>
<td>3.18</td>
<td>3.15</td>
<td>3.18</td>
<td>3.17</td>
<td>3.17</td>
<td>0.01</td>
<td>0.4</td>
</tr>
<tr>
<td><strong>Projected</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall Length</td>
<td>11.70</td>
<td>11.48</td>
<td>11.73</td>
<td>11.69</td>
<td>11.66</td>
<td>11.63</td>
<td>11.52</td>
<td>11.63</td>
<td>0.09</td>
<td>0.8</td>
</tr>
<tr>
<td>Inner Ball Length</td>
<td>8.34</td>
<td>8.15</td>
<td>8.31</td>
<td>8.33</td>
<td>8.30</td>
<td>8.30</td>
<td>8.18</td>
<td>8.27</td>
<td>0.08</td>
<td>0.9</td>
</tr>
<tr>
<td>Outer Ball Length</td>
<td>7.30</td>
<td>7.15</td>
<td>7.35</td>
<td>7.28</td>
<td>7.34</td>
<td>7.28</td>
<td>7.15</td>
<td>7.26</td>
<td>0.08</td>
<td>1.1</td>
</tr>
<tr>
<td>Heel Length</td>
<td>1.55</td>
<td>1.51</td>
<td>1.51</td>
<td>1.54</td>
<td>1.54</td>
<td>1.60</td>
<td>1.52</td>
<td>1.54</td>
<td>0.03</td>
<td>2.0</td>
</tr>
<tr>
<td>Ball Width</td>
<td>4.20</td>
<td>4.26</td>
<td>4.20</td>
<td>4.23</td>
<td>4.22</td>
<td>4.24</td>
<td>4.23</td>
<td>4.23</td>
<td>0.02</td>
<td>0.5</td>
</tr>
<tr>
<td>Heel Width</td>
<td>2.52</td>
<td>2.51</td>
<td>2.49</td>
<td>2.51</td>
<td>2.48</td>
<td>2.50</td>
<td>2.49</td>
<td>2.50</td>
<td>0.01</td>
<td>0.6</td>
</tr>
<tr>
<td>Ankle Width</td>
<td>3.15</td>
<td>3.13</td>
<td>3.14</td>
<td>3.15</td>
<td>3.12</td>
<td>3.15</td>
<td>3.14</td>
<td>3.14</td>
<td>0.01</td>
<td>0.4</td>
</tr>
<tr>
<td>Inner Arch Height</td>
<td>1.28</td>
<td>1.31</td>
<td>1.27</td>
<td>1.25</td>
<td>1.36</td>
<td>1.25</td>
<td>1.24</td>
<td>1.28</td>
<td>0.04</td>
<td>3.3</td>
</tr>
<tr>
<td>Outer Arch Height</td>
<td>0.70</td>
<td>0.64</td>
<td>0.65</td>
<td>0.66</td>
<td>0.63</td>
<td>0.68</td>
<td>0.69</td>
<td>0.66</td>
<td>0.03</td>
<td>4.0</td>
</tr>
<tr>
<td><strong>Circumference</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ball Girth</td>
<td>10.40</td>
<td>10.54</td>
<td>10.37</td>
<td>10.40</td>
<td>10.36</td>
<td>10.43</td>
<td>10.57</td>
<td>10.44</td>
<td>0.08</td>
<td>0.8</td>
</tr>
<tr>
<td>Instep Girth</td>
<td>10.85</td>
<td>10.92</td>
<td>10.84</td>
<td>10.87</td>
<td>10.85</td>
<td>10.84</td>
<td>10.94</td>
<td>10.87</td>
<td>0.03</td>
<td>0.3</td>
</tr>
</tbody>
</table>
the large toe feature, the error may be explained by the inaccurate scaling of the inner toe region during the alignment optimization (Figure 4.7). More importantly, the data sets do not include points around the large toe, as can be seen in Figure 4.7. Therefore, the vertices defining those patches on the surface are not constrained and may move freely to fit the surrounding patches.

Another source of the larger deviations may also explain the errors in the ball lengths and the ball girth measures. Data sets 2 and 7 (the rotated version of set 2) differ in value by as much as 0.2 inches from the other five data sets. Within each of these two groups, the maximum deviations are 0.05 (arch heights and overall lengths for the group of five, and arch heights alone for the group of two). The data sets were compared by overlaying one data set on top of another; however, no noticeable difference is detected which would cause the discrepancy. The remaining measures, shown in Table 4.4, show much smaller deviations.

The accuracy of the feature-based modeling algorithm is tested using Feet 3 and 4, Figures 3.28 and 3.29 respectively, from Chapter 3. Table 4.5 gives seven measures provided by the supplier of the data sets. Also included in the table are the extracted measures, the associated differences and the percentage errors. The girth and projected measures for each data set are displayed in Figures 4.10 - 4.13. As can be seen in the figures, the algorithm correctly matches the model features to those of the data sets. However, the outer ball length of Foot 3, the heel and ball widths of Foot 4 and the instep girths of both show relatively large errors. Because the measurements were taken by hand, the differences may be a result of the subjective selection of the location of the features. Additionally, the fatty tissues of the foot may deform by as much as a quarter of an inch depending on the pressure applied during the hand measurements and the laser scanning. Similar to the six data sets of the previous section, the scans of Feet 3 and 4 fail to capture
the extents of the toes. This results in relatively large differences in the overall length measures since the toe features are unconstrained.

Table 4.5  Supplied and calculated measures for Feet 3 and 4. All measures are in inches. The associated differences and percentage errors are also provided.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Length</td>
<td>10.98</td>
<td>11.26</td>
<td>0.28</td>
<td>2.6%</td>
<td>10.98</td>
<td>11.33</td>
<td>0.35</td>
<td>3.2%</td>
</tr>
<tr>
<td>Inner Ball Length</td>
<td>7.72</td>
<td>7.75</td>
<td>0.03</td>
<td>0.4%</td>
<td>7.95</td>
<td>8.03</td>
<td>0.08</td>
<td>1.0%</td>
</tr>
<tr>
<td>Outer Ball Length</td>
<td>7.05</td>
<td>6.73</td>
<td>0.32</td>
<td>4.5%</td>
<td>6.96</td>
<td>6.99</td>
<td>0.03</td>
<td>0.4%</td>
</tr>
<tr>
<td>Heel Width</td>
<td>2.80</td>
<td>2.78</td>
<td>0.02</td>
<td>0.7%</td>
<td>2.95</td>
<td>2.81</td>
<td>0.14</td>
<td>4.7%</td>
</tr>
<tr>
<td>Ball Width</td>
<td>4.13</td>
<td>4.16</td>
<td>0.03</td>
<td>0.7%</td>
<td>4.37</td>
<td>4.24</td>
<td>0.13</td>
<td>3.0%</td>
</tr>
<tr>
<td>Ball Girth</td>
<td>10.59</td>
<td>10.61</td>
<td>0.02</td>
<td>0.2%</td>
<td>10.74</td>
<td>10.72</td>
<td>0.02</td>
<td>0.2%</td>
</tr>
<tr>
<td>Instep Girth</td>
<td>10.20</td>
<td>11.26</td>
<td>1.06</td>
<td>10.4%</td>
<td>10.35</td>
<td>11.40</td>
<td>1.05</td>
<td>10.1%</td>
</tr>
</tbody>
</table>

Figure 4.10 Girth and point-to-point measures for Foot 3.
Figure 4.11 Projected measures for Foot 3.

Figure 4.12 Girth and point-to-point measures for Foot 4.
4.5 Summary

The feature-based fitting algorithm presented in the previous chapter was tested for data sensitivities, repeatability, accuracy and limitations. Using the slice data set given in Figure 3.20, the two methods for calculating principal axes are tested. Comparing the alignment parameters for both cloud and slice structures, the tetrahedron summation method produces more accurate alignments than the point-mass technique. As expected, the point-mass alignment is sensitive to the data distribution where as the only limiting factor of the volume calculations is the extent to which the slices represent the object’s shape.

The repeatability of the algorithm was tested using six data sets representing the same foot, varying only in the number of points defining the
surface. Both the registration optimization and the feature extraction values show very repeatable results. The largest deviations involved measurements calculated from feature point at the big toe. Since no data points are present at the tip of the big toe, the patches in that region are unconstrained during the tolerance optimization, allowing the surface to move freely. The accuracy tests show similar results except for the instep girth measures. Differing by an inch, the reason for the error may simply be a difference in the definition of the girth measure location.

The data sets used in the repeatability studies revealed a limitation in the registration optimization. Since the data set closely matched the size of the model from the ball points to the heel, the global scaling was unable to correct for the much smaller toe box in the data. The feature-point deformation and vertex optimization routines were unable to correct the problem. Therefore, enhancements must be made to improve the algorithm, one of the topics of the next chapter.
Chapter 5  Summary and Conclusions

As stated previously, there are numerous applications where global information is extracted from a set of discrete data points that represent a known geometric shape. Feature recognition algorithms or interactive software packages may be used to collect the measures directly from the data or from a surface fit to the data. The latter has the advantage of being a much more compact representation and provides a sound basis for further analyses.

Any one of the surface fitting routines mentioned previously may be used to approximate a data set. Unfortunately, these methods are a “blind” fit to the data. Simply reordering the points within the data set may result in a completely different representation. Therefore, researchers, as presented in Chapter 2, propose deforming a previously defined model to fit a data set.

The feature-based modeling algorithm presented in this thesis achieves this by fitting a generic, B-spline model to a supplied data set. The algorithm differs from previously presented methods in that characteristic features are embedded directly within the model. The advantages of the feature-based modeling algorithm are presented in the following section. The limitations and future work required to improve the results are discussed in the second section. Lastly, potential applications are suggested.

5.1  Advantages

The feature-based fitting process differs from current modeling methods in that surface patches are not arbitrarily placed upon the data, as in a least squares fit. Patches on the feature-based model correspond to specific regions in the data set. As a result of the data partitioning, points that may have been
randomly scattered in the original data set are now ordered by the parametric surface of the model. This can produce quick comparisons between different types of imaging techniques by comparing only those points that fall in the same region of interest.

Once fit to a specified tolerance, the model accurately represents the associated data set. This provides a compact representation since the entire model surface can be recreated from a small collection of control vertices (400-450). For laser scan fitting, this can result in compaction of one or two orders of magnitude. Because a previously defined model is being used, the final fit also has a consistent size. The only exception to this is when a hierarchical refinement is required to fit to within a supplied tolerance. In that case, only the patch locations and vertex offsets need to be recorded, a small addition to the model definition.

The characteristic features embedded within the model provide simplified extraction of pertinent information from the data set. Because the model is a parametric definition, new features are easily added as the need for them arises. Moreover, the structured fitting provided by the presented algorithm eliminates the need for refitting the data once new features are added. The new feature is simply incorporated into the generic model and the new measures are immediately available.

5.2 Limitations and Future Work

The goal of the feature-based modeling algorithm is to accurately fit a supplied data set while automatically recognizing characteristic features within the shape. The results of the 2-D pilot study and 3-D surface implementation show that the method presented adequately meets these objectives. However, there are a number of aspects of the algorithm that require improvement.
5.2.1 Alignment

For non-symmetric objects, center of mass and principal axes alignment provide a good initial orientation for optimization. However, near symmetric objects require a more intelligent method for aligning the model to the data. The data sets used in the repeatability study are nearly symmetric about two axes; a result of scanning further up the leg. The algorithm required a check for max/min extents along the axes to select the proper correspondence between data and model axes.

Articulated models, such as the foot model presented, pose a special problem to the alignment process. Variability in ankle and toe angles may adversely affect the alignment optimization since it is simply minimizing the squared errors in the modeled position. To simulate these articulations, the model may be embedded within a free-form deformation (FFD) matrix [LAMO94, SEDE86] and rotational transformations may be applied to the appropriate regions [BARR84]. Once implemented, the FFD mesh could also be used to scale sections of the model separately. For example, the toe box of the model in Figure 4.7 could be scaled separately from the rest of the foot to better fit the data.

5.2.2 Feature Deformation

For data fitting, one of the main advantages of B-spline surfaces is local manipulation. However, this property can become a disadvantage during the feature deformation routine. As shown in Figure 4.8, the solution for the feature point displacement creates localized peaks, valleys or folds on the surface. In the worst case, subsequent vertex optimization may create creases or folds, destroying the smooth surface. The preferred effect is a broad movement of the patches surrounding the feature points, which may be
achieved by embedding the model within an FFD matrix. A properly defined FFD grid implemented in the component alignment algorithm may also be used in this respect.

Currently, an averaged error vector for each feature point drives the feature deformation algorithm. This requires that the closest point parameters of at least one data point fall within each feature point range. Since no data points fall within the ranges of the tip and side of the big toe in Figure 4.7, those feature points do not move, resulting in a poor fit. To improve the results of the feature fit, more intelligence may be incorporated into the closest point optimization to more accurately associate data points to the feature ranges. In addition, the feature-point deformation routine may be controlled by an optimization instead of the calculated average vectors.

5.2.3 Vertex Optimization

One objective of the feature-based modeling algorithm is to accurately represent the shape of a supplied data. Even though the final approximation may fit a data set to a tolerance, the shape of the model may include extraneous characteristics not representative of the underlying generic shape of the object. For example, the upper heel patches on the optimized surfaces in Figures 3.27 and 3.28 fold into the model. Since the data sets do not contain points in these regions, the vertices defining the patches are free to move to better fit the neighboring patches with data points. To keep the inherent shape in these regions, patches with no associated data points should be constrained by either limiting the movement of their vertices or by adding data in these areas.

A side effect of the latter corrective measure is that the vertex optimization must handle more points than originally contained in the data set. Already, the algorithm does not converge as rapidly as desired. To improve the
performance of the algorithm, a Newton's method with single iteration conjugate gradient system solution may be implemented. The advantages of such an optimization are not known since the second order derivatives must be calculated and the gradient-based linear system approximation must be near the minimum for the optimization to converge.

Another problem with the optimization is slow convergence near the minimum. To improve the overall efficiency of the algorithm, the optimization could be stopped at a larger functional tolerance. Spans containing points that are out of the specified data tolerance may then be subdivided using hierarchical B-splines. Since a single hierarchical refinement only allows one vertex to move, the optimization of its position and weight is extremely simplified. Unfortunately, the refinement additions would increase the size of the model slightly. Thus, a balance must be found between the convergence of the algorithm and the size of the resulting model.

5.2.4 Miscellaneous Additions

As stated previously, one of the advantages of the feature-based algorithm is that the surface patches of the model correspond to specific features on the object. Different data tolerances may be associated with areas of high or low levels of detail. For example, the jowl and upper forehead spans of the 2-D facial profiles could have larger tolerances than the nose or mouth regions. This would prevent hierarchical implementations in regions where the added detail is not necessary.

Using the surface normals, the patch-based tolerances could be extrapolated to interior and exterior tolerances as well. Additionally, the normals could be used to penalize interior or exterior errors within the minimization cost functions. In this manner, the model may be shrunk around
the data from the outside or ballooned from within the data. This technique alone may correct the misalignment shown in Figure 4.7 by penalizing internal errors, forcing the model to be contained within the data set.

Both the 2-D and the 3-D implementations of the feature-based fitting algorithm use models with free/open ends. Nothing restricts the amount of data collected beyond these ends which the algorithm will normally try to fit. For example, the four data sets shown in Figures 3.24 and 3.27-3.28 vary greatly in the amount of data collected above the ankle. Though not implemented, the algorithm could easily ignore any data points whose closest point parameters lie along the open end of the model. This would eliminate the stretching necessary to fit these outlying points which restricts the alignment and fitting capability of the model.

Because the feature parameters are held constant during the point deformation routine, the final features have the same parametric location on every model after being fit. This property alone provides unique opportunities for model-to-model or data-to-data comparisons. However, the repeatability and accuracy of the algorithm may be improved by adjusting the location of the feature points after the final fit. A parameter optimization routine could be implemented to ensure the proper placement of the features relative to the defined foot axes.

5.3 Potential Applications

Since the feature-based modeling has such a generalized formulation, the algorithm can be readily implemented in a number of applications. For reverse engineering, feature-based models may be used to recover manual modifications of a part into an existing CAD design. Since quality control
requires measurements from numerous parts of the same shape, the algorithm presented may be ideal for automated testing.

Though applicable to industry, feature-based models may be more advantageously applied within the medical field. Proper diagnosis and treatment of disease frequently requires the measurement of anatomical features. For example, radiation dosages for cancer treatment rely heavily on the accurate measurement of surface area in the affected regions [ALLA93]. Simply summing the areas of discretized patches in a model would provide this measure. The feature-based algorithm may also prove advantageous in surgical planning and the construction of prototype implants [LAVA96b, MAHO95, PUTT93, VANN84].

Geometric and alignment algorithms are currently being used to track changes in the body due to disease of healing [ALPE90, MCPH96]. Huang, et al. present a registration routine for comparing two data sets to assess and track asymmetry in the human back and torso resulting from scoliosis [HUAN96]. Similarly, Zuk and Atkins register voxel data for use in image-guided neurosurgery and radiotherapy [ZUK96]. Imaging of bones can determine the volumetric changes in grafts as well as check for advanced stages of disease [HAUS90, TOEN90]. In all cases, feature-based models could be applied to accurately measure features within a collected data set, providing a basis for comparative studies across modalities (CT, MRI, or ultrasound scans) [CARL92, VAND93].

An immediate application involves a series of studies funded by the Department of Veterans Affairs for custom construction of prosthetic sockets [BLOC85, BOON89, HOUS92, INMA59, LORD88, MEEK90, NEWY90, SCUR93, SIDL89, TOPP90, TORR92]. The research attempts to overcome the
shortcomings of the artisan without forfeiting the years of acquired knowledge. The process digitizes a plaster cast of the residual limb. The surface points collected are imported into a computer-aided design package for use in manually deforming a socket model. The cited advantages of this procedure are a uniform and reproducible product, compact storage (of the finished model) and accelerated data transmission. These are all advantages provided by feature-based models, but without the need for manual deformations in an interactive system. Similarly, the algorithm could be used for custom clothing and footwear [BAO89, BA092, LORD88, LORD91, RASD90, ROSS93].

Surface approximation and feature extraction from a data set are a crucial component of many applications. The currently available methods for performing these tasks vary greatly. However, few take advantage of the known shape of the object being fit. Ignoring this fact limits the efficiency and the usefulness of the resulting surface approximation. The feature-based algorithm presented in this thesis takes full advantage of the underlying shape as well as the characteristic features that distinguish the surface. Once the model is fit to tolerance, the desired measures are readily available. The algorithm provides a compact and consistently sized representation, independent of the data set orientation, density or resolution. The solution provides order to a randomly dispersed point cloud, conveniently allows model-to-model or data-to-data comparisons and provides a means for tracking changes within or building a statistical database for collections of data sets. Already, the foot model implementation extracts more information about the foot than currently used procedures.
Bibliography


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


Appendix: Additional Data Sets

The following data sets are used to test the feature-based fitting algorithm. All of the sets were supplied courtesy of a joint venture of Laser Design, Inc. and Red Wing Shoe Company. The scanner is a proprietary product of the two companies. Unfortunately, the exact measurements for each data set were not provided.
Figure A.1 Version number 1 of the data set used in the feature deformation, tolerance fitting and repeatability testing (3,109 data points).
Figure A.2 Version number 2 of the data set used in the feature deformation, tolerance fitting and repeatability testing (4,713 data points).
Figure A.3  Version number 3 of the data set used in the feature deformation, tolerance fitting and repeatability testing (17,091 data points).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure A.5  Version number 5 of the data set used in the feature deformation, tolerance fitting and repeatability testing (9,965 data points).
Figure A.6 Version number 6 of the data set used in the feature deformation, tolerance fitting and repeatability testing (5,179 data points).
Vita

Greg was born on August 11, 1968, in New Orleans, Louisiana, to Earl and Judy Dobson. He attended Brother Martin High School and graduated in 1986. Greg was awarded a bachelor of science degree in Mechanical Engineering from Louisiana State University (L.S.U.) in 1990. Immediately following graduation, Greg entered the graduate program at Louisiana State University and was awarded the Board of Regents Dean’s Fellowship. His areas of concentration were mechanical design and computer-aided geometric modeling. In June 1995, Greg married L.S.U. Alumnus Lisa Carroll, who motivated the completion of his doctoral research in October 1997. At the December commencement, Greg received his doctor of philosophy degree.
DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Gregory Todd Dobson

Major Field: Mechanical Engineering

Title of Dissertation: Feature-Based Models for Three-Dimensional Data Fitting

Date of Examination: October 24, 1997

Approved:

[Signatures]

Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

[Signatures]

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.